American Put Option

- An American Call (resp. Put) option on a stock gives the right to buy (resp. Sell) the stock at any time between now and time *T* for a pre-agreed price *K*
- It is proven that, if the stock does not pay dividends (more generally, if the drift of the GBM is larger or equal to the risk free rate), it is never optimal to exercise early. Therefore it has the same value as a European claim
- Nothing can be said about the Put, and there are circumstances where it might be convenient to exercise early
- Intuitively, imagine the spot falls well below the strike K, near to zero. If we hold the option, the price could fall even lower, but there is limited upside, because the spot cannot drop below zero, so the “time value” of the option is limited. This might be less then the money we could make if we exercise the option and invest now the proceedings K-S at the risk free rate until maturity.

American Put Option

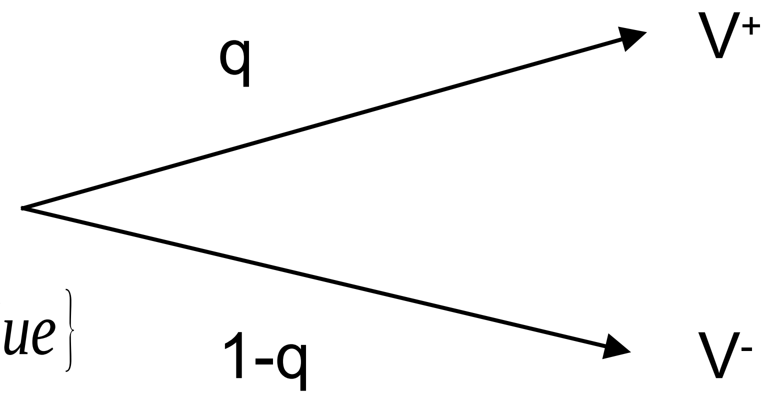
- The binomial tree is quite effective in pricing American contingent claims.
- We cannot consider only the last time period, as we did for the European option, because we are interested also in opportunities to **exercise early**
- The procedure is quite simple: we compute the expectation backward in the tree, but, at every node, we compare the expected value obtained with the value of exercise immediately, and we choose the largest of the two

American Put Option Algorithm

- Initialize the value of the derivative at the last period of the tree, using the payoff function $f(S)$
  - Compute the discounted value (E) at the previous period
  - Replace the discounted value at the previous period with max[E, $f(S)$]
  - If not time 0, then go to previous period

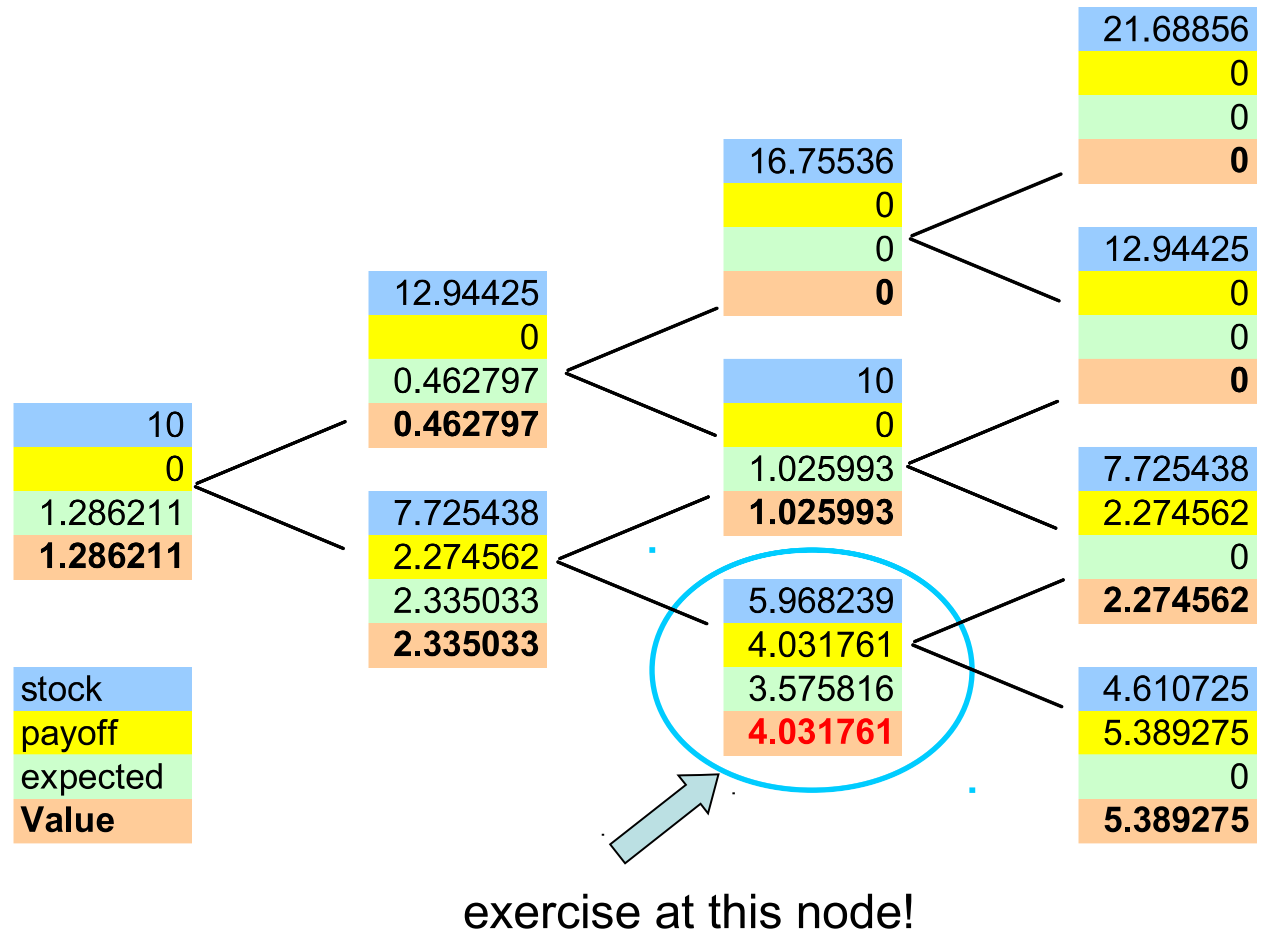
American Put Option Algorithm

$$Cont.Value = e^{-r\Delta t}\left[qV^{+} + (1-q)V^{-}\right]$$
$$Exer.Value = f(S)$$
$$Deriv.Value = \max\{Cont.Value, Exer.Value\}$$



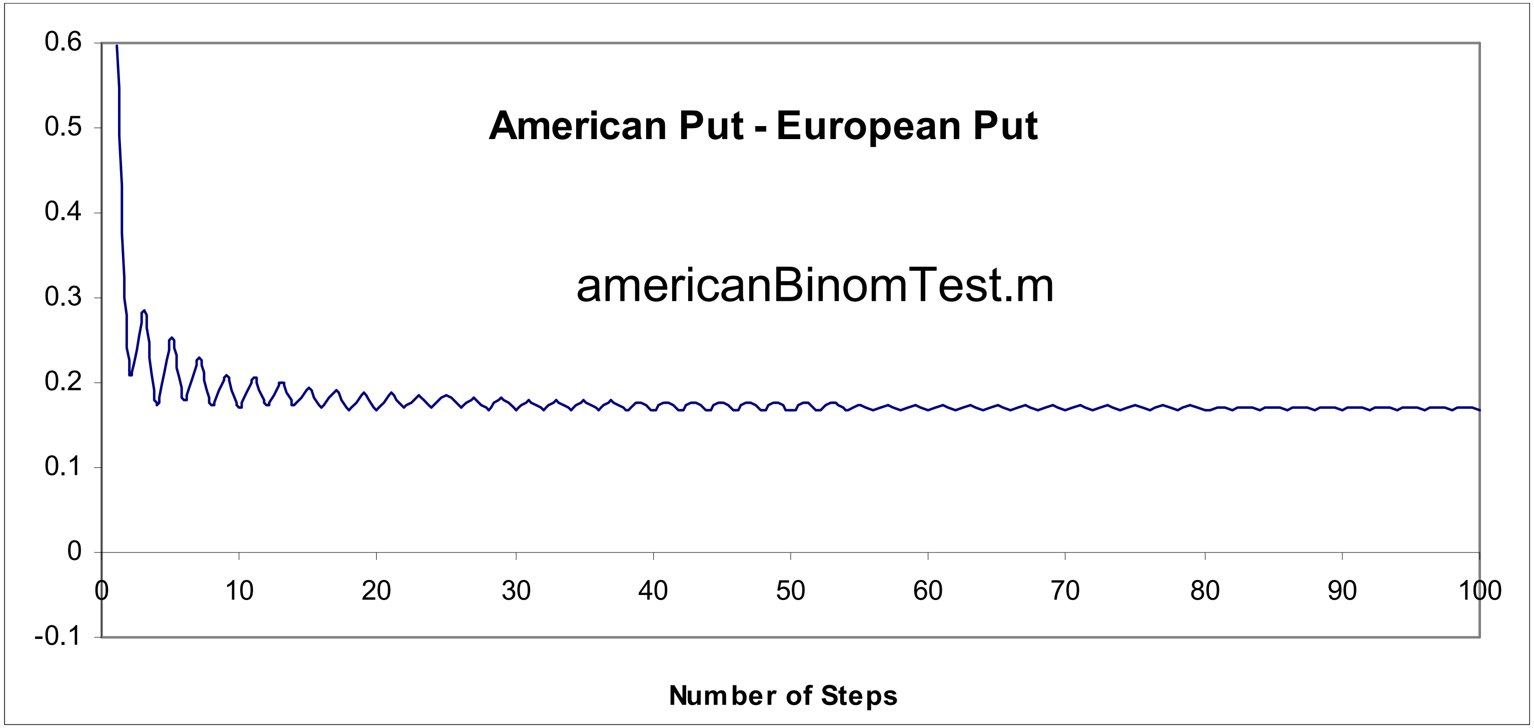
American Put Example

N	3
T	2
Spot	10
Strike	10
RiskFree	7%
Volatility	0.3
d	0.772544
u	1.294425
p	0.527378
df	0.954405



American Put Convergence

- The following chart shows the binomial tree approximation minus the exact price of a European put as a function of the number of periods in which we subdivide the 2 years
- To put the convergence pattern in perspective, the European Put price is 1.000



American Put Algorithm

- Because now we need the entire tree, one may think that we need to keep track of lot of numbers (allocate enough memory to keep all nodes of the price tree and of the derivative tree)
- In reality, with some ingenuity, we can use one single vector of numbers, and keep rewriting on it

American Put Algorithm

Step 4	Step 3	Step 2	Step 1	Step 0
$f(d^4S)$				
$f(d^3uS)$				
$f(d^2u^2S)$				
$f(du^3S)$				
$f(u^4S)$				

American Binomial Pricer

```
function val = americanBinomialPricer( d, u, p, T, N, rf, spot, payoff )

          ratioUD = u / d;
          df = exp( -rf * T / N );
          q = 1.0 - p;

          v = zeros( N + 1, 1 );

          price = spot * d ^ N; % price corresponding to state j = 1
          lastLowest = price; % lowest price in this time step

          for j = 1:N+1
                    v( j ) = payoff(price);
                    price = price * ratioUD;
          end

          for tS = N:-1:1
                    lastLowest= lastLowest / d;
                    price = lastLowest;
                    for j = 1:tS
                              v(j)= max( payoff(price), df * ( p * v(j+1) + q * v(j) ) );
                              price = price * ratioUD;
                    end
          end

          val = v( 1 );
```

American Binomial Pricer (2)

- We can eliminate the inner loop using Matlab vectorial operation, which yields a more compact elegant and faster code

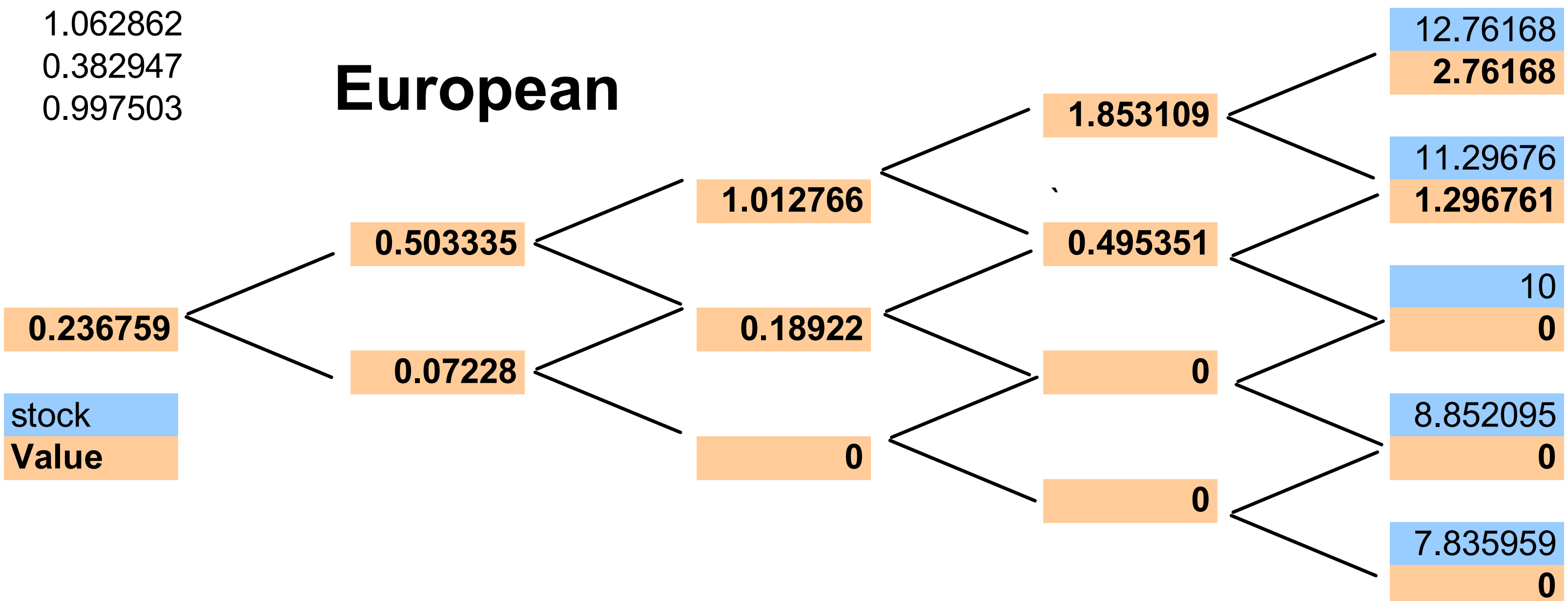
```
function val = americanBinomialVec( d, u, p, T, N, rf, spot, payoff )

          ratioUD = u / d;
          df = exp( -rf * T / N );
          q = 1.0 - p;

          v = zeros( N + 1 );

          lastLowest = spot * d ^ N;
          prices = lastLowest * ((ratioUD * ones( N + 1, 1 )) .^ (0 : N)');

          v = payoff(prices);

          for tS = N:-1:1
                    prices = prices( 1:tS ) ./ d;
                    v = max( payoff(prices), df .* ( p .* v( 2:tS+1 ) + q .* v( 1:tS ) ) );
          end

          val = v( 1 );
```

American Call With Dividends

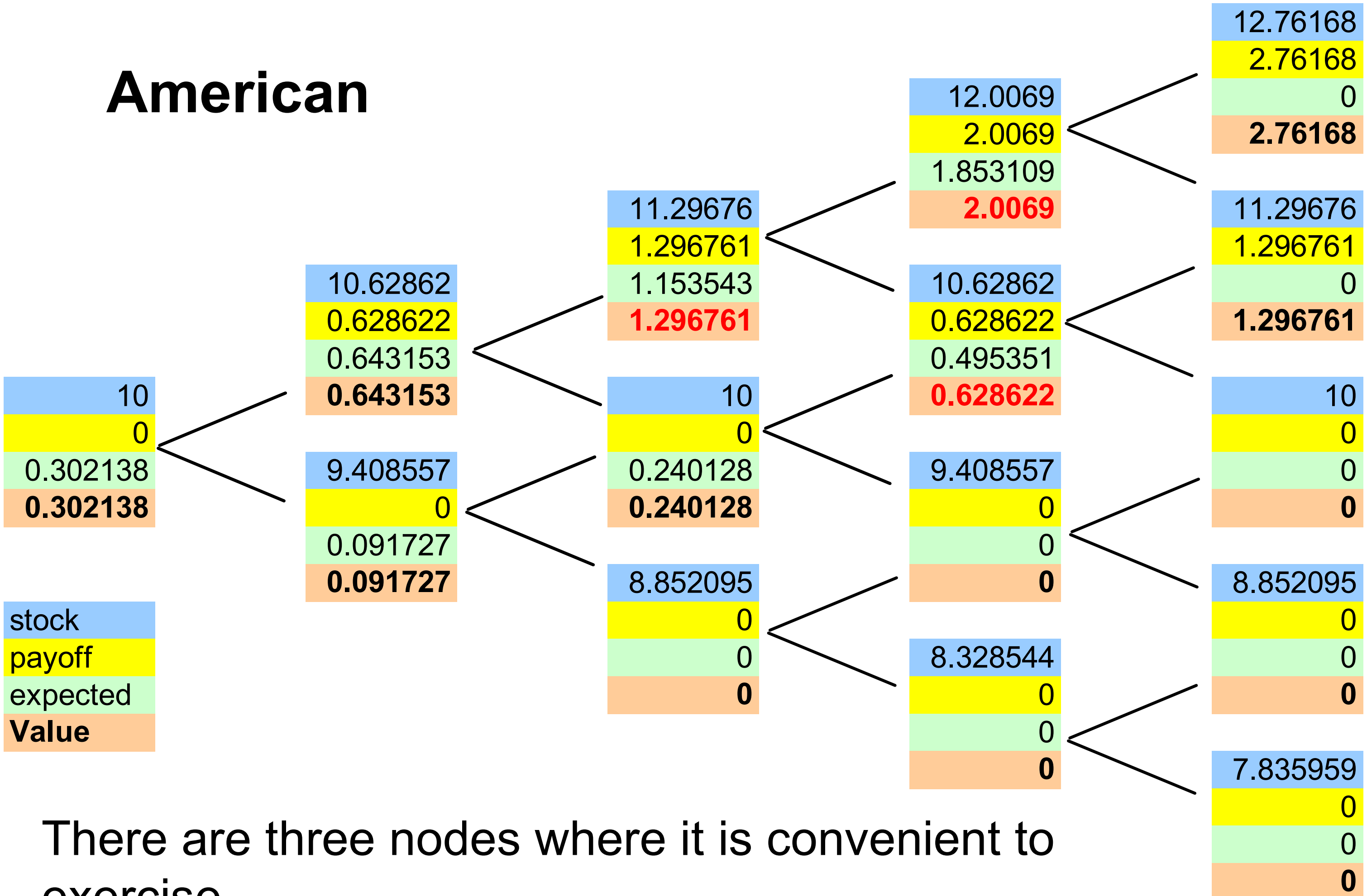
N	4
T	1
Spot	10
Strike	10
RiskFree	1%
Yield	6%
Volatility	12%
d	0.940856
u	1.062862
p	0.382947
df	0.997503

- Note that in presence of dividends it may be optimal to exercise early an American call
- Let's compare the valuation with a European Call

European



American Call with Dividends



- There are three nodes where it is convenient to exercise
- Much more value than in the European case

American vs European Call

- Difference is larger when S/K is large or r/yield is low
- The tree gives both the optimal exercise strategy and the correct hedge ratios

Bermuda Options

- Bermuda can be exercised only at certain dates
- That is easy to take into account in a tree (just do the check for early exercise at time steps where allowed only)
- If the exercise dates do not match exactly the time steps of the tree, usually assume that exercise is possible at the closest time step (ok if time steps are small)

Compounded Options

- A compounded is such that I can make several decision during the life of the contract, and every decision made at time *t* affects the level of flexibility available at times after *t*
- For example a European option expiring at time *t1* to buy a European option expiring at time *t2* at a pre-agreed premium
- This is **easy** to price on a tree. For instance, for this example, just proceed in two steps
  1. Value backward the second option on the tree from time t2 to time t1, and save its values at every node
  2. Value the first option from time t1 to time t0, using the price of the first option at time t1 as the underlying

Compounded Option Example

American Option K=0.5 USD, T=0.5y

on an

American Option K=10, T=1y

RiskFree	1%
Yield	6%
Volatility	12%
d	0.940856
u	1.062862
p	0.382947
df	0.997503

Price Tree

				12.762
			12.007	
		11.297		11.297
	10.629		10.629	
10.000		10.000		10.000
	9.409		9.409	
		8.852		8.852
			8.329	
				7.836

American Call @ 10

				2.762
			2.007	
		1.297		**1.297**
	0.643		**0.629**	
0.302		0.240		0.000
	0.092		0.000	
		0.000		0.000
			0.000	
				0.000

American Call @ 0.5

		0.797	=max(1.297-0.5,0)
	0.304		
0.116		0.000	
	0.000		
		0.000	

max(0.643-0.5, 0.997*(0.383*0.797+(1-0.383))*0)

Summary

- Steps:
  1. Calibrate the parameter of a stochastic models to the price of observed tradeables (at least forwards, which will make the model risk neutral)
  2. Choose an appropriate state variable
  3. Construct a tree describing as accurately as possible the dynamic of the state variable (both globally and locally)
  4. Price the derivative
- Note that tree construction and pricing are fairly orthogonal problems, although sometimes knowledge of the pricing problem may influence construction of the tree grid (e.g. to place a strike with respect to the tree nodes)

Greeks Estimation

- In the binomial world delta can be obtained very quickly from the tree itself
- Because the tree is based on a non-arbitrage argument, we have an explicit formula which computes delta at every node
- And there are also some methodologies which apply to special cases, but not generalizable
- The most common technique is to use Finite Differences approximation, i.e. we perturbate some parameter of the model and we reconstruct the tree

Greeks Estimation

- In these formulas *x* here can be any of the model parameters (e.g. spot price, interest rate, volatility,…)
- We classify according with difference type and bump type
  - Forward, central or backward differences
  - relative or absolute bump

	absolute	relative
forward	$\dfrac{f(x+h)-f(x)}{h}$	$\dfrac{f(x(1+h))-f(x)}{xh}$
central	$\dfrac{f(x+h)-f(x-h)}{2h}$	$\dfrac{f(x(1+h))-f(x(1-h))}{2xh}$
backward	$\dfrac{f(x)-f(x-h)}{h}$	$\dfrac{f(x)-f(x(1-h))}{xh}$

Central vs Left/Right

- Central differences are usually more accurate than forward or backward differences (as we can show via Taylor expansion), but they are also more expensive to compute (they requires two extra valuations)
- Care needs to be taken not to introduce unreasonable values (e.g. a correlation larger than 1, or a negative volatility), in which case sometime we may prefer or fallback on one-sided (left or right) differences, instead of central

Relative vs Absolute Bump

- The choice of absolute vs relative bump usually depends on the characteristics of the variable at hand
- If we want the sensitivity with respect to a price, whose level can vary wildly from one asset to another (e.g. EUR price is 1.3 USD, while GOLD price is 1500 USD), we normally use **relative** bumps
- For sensitivities with respect to variables which are usually within certain ranges (e.g. interest rates, volatilities, correlations), usually we use **absolute** bumps

Path Dependent Derivatives

- In a path-dependent derivative the payoff depends on the path followed by the price of the underlying asset, not just its final value.
- I.e. the derivative is a function of t, S_t and some function f($\bullet$) of the historical values assumed by S_t in the time interval $t \in [0, T]$
- The dependency on past fixing could be *discrete* (e.g. the daily "close" value of the stock) or *continuous* (e.g. the maximum reached by the stock price anytime during the day)

Path Dependent Derivatives

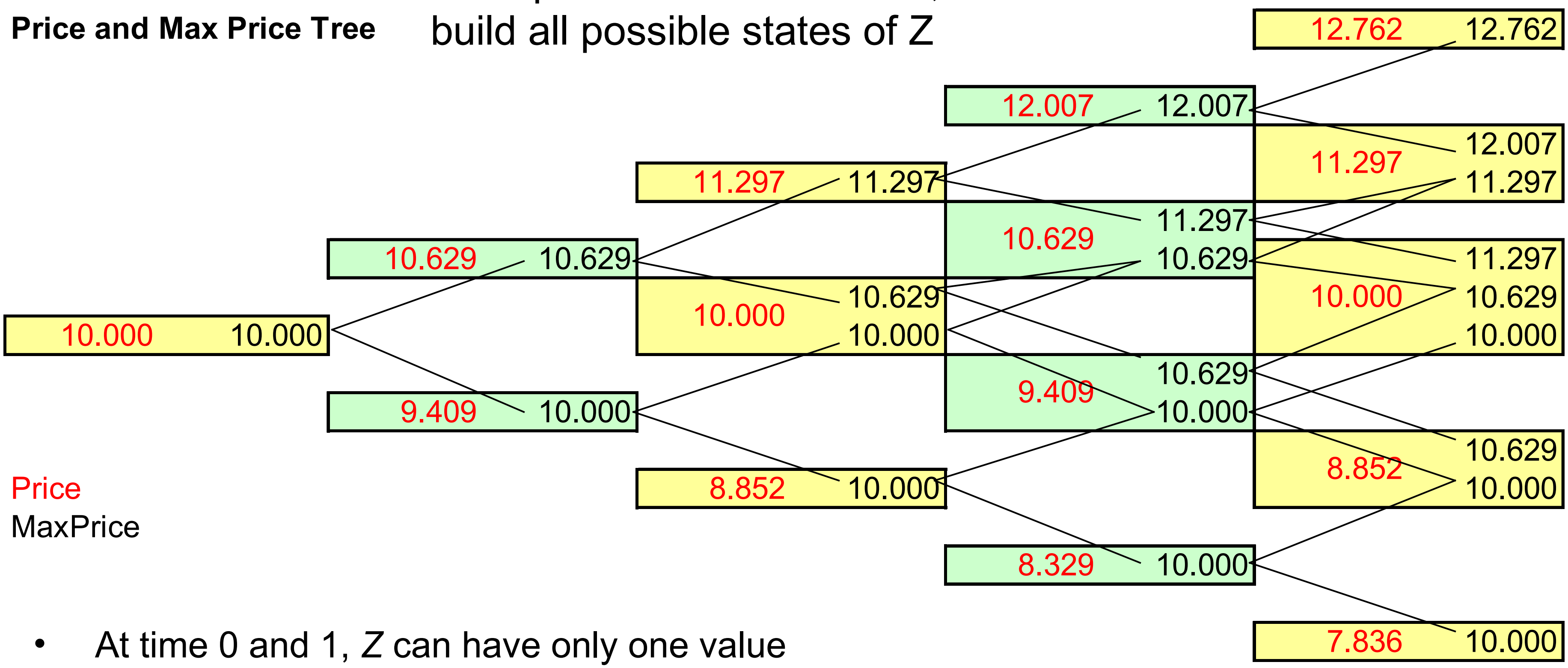
- A technique called **Forward Shooting Grid** (Hull, White 1993) can be used to price these options on a tree.
- It introduces an auxiliary state variable, which keep track of the possible values of the function f(•) in the different states of the tree
- For efficiency, it is important that:
  - The payoff from the derivative must depend on a single function, f(•), of the path followed by the underlying asset.
  - It must be possible to calculate the updated value of f(•) at time $t+\Delta t$ from the known value of f(•) at time t and the updated value of the underlying asset at time $t+\Delta t$.
  - The number of possible values of the function f(•) does not grow too much on the tree

American Lookback Option Example

- The payoff of this particular lookback option is: $\max[S^{MAX}(T)\text{-}S(T),0]$, where $S^{MAX}(T)$ is the maximum value assumed by S_t over in the interval $t \in [0,T]$
- We can introduce in the tree the auxiliary variable $Z_t = \max[S_t]$ t $\in [0,t]$
- If we move from the state (t, S_t, Z_t), to a new state $(t+1, S_{t+1})$, Z_{t+1} will take the value $\max[Z_t, S_{t+1}]$

American Lookback Option Example

Sweep the tree FORWARD, to build all possible states of Z



- At time 0 and 1, *Z* can have only one value
- At time 1 the central node can be reached by two different nodes, and Z has two possible values
- At time 2, in the two central nodes Z has two possible values
- At time 3, in the central node, Z can have 3 possible values
- Note at any node Z has at least value 10, which is the initial value

American Lookback Option Example

Payoff Value and Derivative Value Tree

								0.000	0.000
						0.437	0.000		
								0.710	0.710
				0.658	0.000			0.000	0.000
						0.798	0.668		
		0.848	0.000			0.387	0.000	1.297	1.297
				0.969	0.629			0.629	0.629
0.976	0.000			0.583	0.000			0.000	0.000
						1.334	1.220		
		1.059	0.591			0.707	0.591		
								1.777	1.777
				1.360	1.148			1.148	1.148
						1.770	1.671		
								2.164	2.164

Derivative Value
Exercise Value

- When working backward, need to compute the value of the derivative for every possible Z
- Need to pick the correct values from the next period, as shown from the arrows in the previous Figure

American Lookback Option Example

Stock	Z	Payoff	Expect	Value
10.629	11.297	0.668	0.798	**0.798**
	10.629	0.000	0.387	**0.387**

Stock	Z	Value
11.297	12.007	**0.710**
	11.297	**0.000**
10.000	11.297	**1.297**
	10.629	**0.629**
	10.000	**0.000**

- Focusing on one of the node (t=3,j=2) of previous figures:
  - Price is 10.629 and can move either up to 11.297 or down to 10
  - Z has two possible states: 10.629 and 11.297.
    - If the price moves to 11.297, the new state of Z is 11.297, regardless of the current state.
    - If the price moves down to 10, then Z preserve its current value
  - When we compute the value of the node corresponding to the two possible values of Z
    - take the expectation of the correct nodes at time t+1
    - compare with the value of exercising at this node, and take the maximum

Path Dependent Options

- Let's formalize the argument
  - Let Z_t be one particular value that the function $f(\bullet)$ can take at time t
  - Let V(n,j,k) denote the option value at the n^{th} time step, at the j^{th} upward jumps from the initial asset value in a binomial model and at the k^{th} state of the various possible values of the state variable Z_t
  - Let's define the state transition function G: $Z_{t+\Delta t}=G(t,S_{t+\Delta t},Z_t)$, or, more simply its discrete equivalent $g(n,j,k)$

$$V(n,j,k)=e^{-r\Delta}\left[p\cdot V\left(n+1,j+1,g(n,j+1,k)\right)+(1-p)\cdot V\left(n+1,j,g(n,j,k)\right)\right]$$

Asian Strike Option Example

- Let's consider an Average Strike option, which has payoff=max[S_T - Avg{S_t:$t \in [0,T]$}, 0]
- Unfortunately, the number of possible values for the averaging value Z at a binomial node for the arithmetic averaging option grows exponentially at 2^n, where n is the number of time steps from the tip of the binomial tree. (Why 2^n? Since there are 2n possible realized asset paths after n time steps and each path gives a unique arithmetic averaging value)

Asian Strike Option Example

Price Tree

			12.007
		11.297	
	10.629		10.629
10.000		10.000	
	9.409		9.409
		8.852	
			8.329

T	0.75
Spot	10
RiskFree	0.01
Yield	0.06
Volatility	0.12

Average Tree

			10.983
		10.642	
			10.639
	10.314		10.314
		10.210	10.009
10.000		9.803	
			10.009
	9.704		9.704
			9.417
		9.420	
			9.147

$Z(ddu)=(10+9.409+8.852+9.409)/4$

- At the third step, the average can be in 8 different states
- At every node we have (N,j) possible values
- Note that these arithmetic average values do not coincide with the stock prices at the corresponding nodes

Asian Strike Option Example

- A possible remedy is to restrict the possible values for the average Z to a certain set of predetermined values.
- The option value $V(S, Z, t)$ for other values of Z is obtained from the known values of V at the chosen predetermined set of Z values by an interpolation between the nodal values
- The methods of interpolation can vary. E.g., we could use nearest node interpolation, linear (between 2 neighboring nodes) and quadratic interpolation (between 3 neighboring nodes).

Asian Strike Option Example

- How do we predetermine which values of the average Z we want to use?
- We could use:
  - a common grid across all nodes and time steps
  - a common grid across all nodes in the same time step
  - have different grid for every node
- I chose here the first approach, which is the easiest to implement, but still useful to understand the method and to illustrate a few tricks we can use in designing the algorithm

Asian Strike Option Example

- To define a discretization step for the variable *Z*, we look at the minimum and maximum values that the average can take at the last time step

$$minAvg=\sum_{n=0}^{N} S_0 d^n, \qquad maxAvg=\sum_{n=0}^{N} S_0 u^n$$

- We subdivide this interval in a number of sub-intervals, of equal size in log-space, proportional to the number of nodes in the last interval *N*+1 (other choices might be suitable, or even be superior)
- For instance, we could choose to have *N*m* sub-intervals (with m>1)
- The largest we choose m
  - the more accurate the result will be
  - the more expensive computations will be

Asian Strike Option Example

- With *m*=2, since N=3, we should have 6 sub-intervals, equally spaced in log-space between 8.934 and 10.983

$$logStep=\frac{1}{Nm}\ln\frac{maxAvg_N}{minAvg_N}=\frac{1}{6}\ln\frac{10.983}{9.147}=0.03441$$

- We index the values of *Z*, with reference to the price at the top of the tree, which has index zero

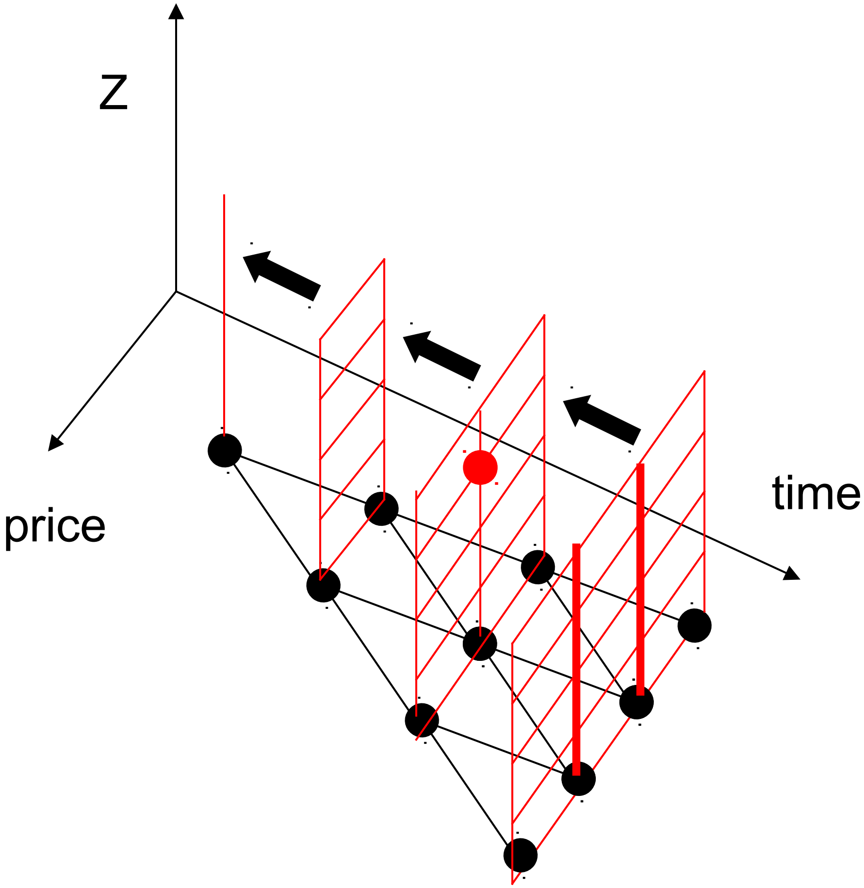
$$Z_k=S_{0,0}e^{k\cdot logStep}$$

- where *k* belongs to a range sufficient to cover the maximum and minimum values of the average achievable (we'll come back to this)

Asian Strike Option Example

- At every time step now we need to compute the value of the derivative in correspondence to different values of Z, for each possible state of the price S
- It means we have now a matrix of values of the derivative, not a vector as we had before, i.e. what gets propagated back trough the tree is a matrix, not a vector
- In general, the size of the matrix at step n is: $(n+1)*(Nm+1)$

Asian Strike Option Example



- Consider the value of the derivative marked with a red dot. It depends on the values of the derivative on “some” points on the two bold red lines, but it is unlikely it depends exactly on the points on the grid, therefore interpolation is needed
- The values on the red lines should cover a range of values sufficient to allow to resolve all interpolation queries arising from the computation of the value of all points in the previous grid
- We need to make sure that is the case!

Asian Strike Option Example

- We determine the value of the minimum k_{min} and maximum k_{max} value of the index *k* at every possible node such that the minimum and maximum values of the average achievable at this node are strictly contained in the interval implied
- To do that, we propagate forward through the price tree
- At node (*n*,*j*)

$$k_{min}(n,j)=\text{floor}\left(\frac{1}{logStep}\ln\frac{n\cdot e^{logStep\cdot\min\{k_{min}(n-1,i):(n-1,i)\text{ father of }(n,j)\}}+S_{n,j}/A_{0,0}}{n+1}\right)$$

$$k_{max}(n,j)=\text{ceil}\left(\frac{1}{logStep}\ln\frac{n\cdot e^{logStep\cdot\max\{k_{min}(n-1,i):(n-1,i)\text{ father of }(n,j)\}}+S_{n,j}/A_{0,0}}{n+1}\right)$$

Asian Strike Option Example

Price Tree

			12.007
		11.297	
	10.629		10.629
10.000		10.000	
	9.409		9.409
		8.852	
			8.329

logStep	0.030483

Min Max Average Index

			4
			3
		3	
		2	
	2		3
	1		-1
0		2	
0		-1	
	0		2
	-1		-2
		-1	
		-2	
			-2
			-3

Z grid

k	Z
5	11.646
4	11.297
3	10.958
2	10.629
1	10.310
0	10.000
-1	9.700
-2	9.409
-3	9.126
-4	8.852
-5	8.586
-6	8.329

$$\max\left\{k_{min}(3,1):(2,i)\text{ father of }(3,1)\right\}=\max\{2,-1,2\}=2$$

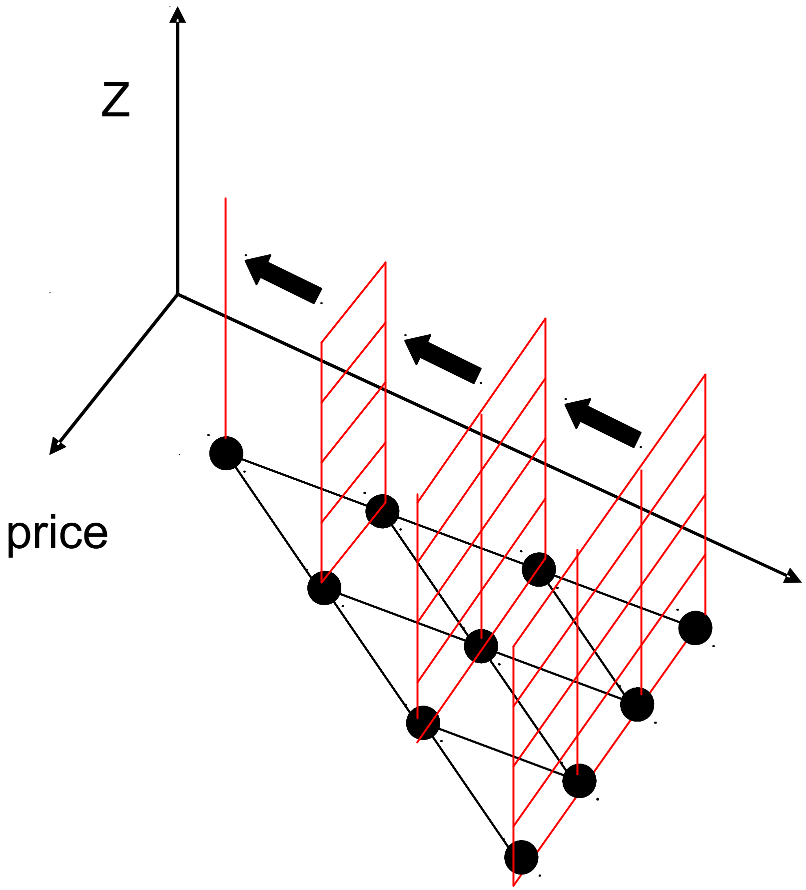
$$AvgMax(2,i)=\frac{3\cdot 10\,e^{0.0305\cdot 2}+9.409}{4}=10.324$$

$$k_{max}(3,1)=\text{ceil}\left(\frac{1}{0.0305}\ln\frac{10.324}{10}\right)$$

Min and max k indices ever: -3,+4

- This tells us exactly which values of *Z* is useful to compute for each price node (*n*,*j*)

Asian Strike Option Example



- The largest size of the matrix is at time 3: 4x8
- Note that the dimension of the matrix which is propagated backward shrink by 1 at each time step in the price direction
- In the Z direction, although we drew the entire matrix, on each vertical line we know exactly which are the values that we need to compute

Asian Strike Option Example

- At the last time step, we just need to initialize the values of the derivative in each useful cell of the matrix, computing the payoff

max[10.629-9.700, 0]

Derivative value at n=3

		k	-3	-2	-1	0	1	2	3	4
min K	**max K**	**price \ Z**	9.126	9.409	9.700	10.000	10.310	10.629	10.958	11.297
3	4	12.007	---	---	---	---	---	---	1.049	0.710
-1	3	10.629	---	---	0.929	0.629	0.319	0.000	0.000	---
-2	2	9.409	---	0.000	0.000	0.000	0.000	0.000	---	---
-3	-2	8.329	0.000	0.000	---	---	---	---	---	---

Asian Strike Option Example

Derivative value at n=3

		k	**-3**	**-2**	**-1**	**0**	**1**	**2**	**3**	**4**
min K	**max K**	**price \ Z**	9.126	9.409	9.700	10.000	10.310	10.629	10.958	11.297
3	4	12.007	---	---	---	---	---	---	1.049	0.710
-1	3	10.629	---	---	0.929	0.629	0.319	0.000	0.000	---
-2	2	9.409	---	0.000	0.000	0.000	0.000	0.000	---	---
-3	-2	8.329	0.000	0.000	---	---	---	---	---	---
Derivative value at n=2										
2	3	11.297	---	---	---	---	---	0.395	0.301	---
-1	2	10.000	---	---	0.266	0.180	0.091	0.000	---	---
-2	-1	8.852	---	0.000	0.000	---	---	---	---	---

- At step n=2 we need to interpolate. For instance, we show how to compute the derivative value circled in red, corresponding to Z=9.700 and S=10
  - At step n=3 S can jump from 10.0 up to 10.629 and the average would become (9.700*3+10.629)/4=9.932, or down to 9.409 and the average would become (9.700*3+9.409)/4=9.627
  - $V(10,9.662,2)=e^{-r\Delta t}[pV(10.629,9.932,3)+(1-p)V(9.409,9.627,3)]$

Asian Strike Option Example

Derivative value at n=3

		k	-3	-2	-1	0	1	2	3	4
min K	**max K**	**price \ Z**	9.126	9.409	9.700	10.000	10.310	10.629	10.958	11.297
3	4	12.007	---	---	---	---	---	---	1.049	0.710
-1	3	10.629	---	---	0.929	0.629	0.319	0.000	0.000	---
-2	2	9.409	---	0.000	0.000	0.000	0.000	0.000	---	---
-3	-2	8.329	0.000	0.000	---	---	---	---	---	---

Derivative value at n=2

2	3	11.297	---	---	---	---	---	0.395	0.301	---
-1	2	10.000	---	---	0.266	0.180	0.091	0.000	---	---
-2	-1	8.852	---	0.000	0.000	---	---	---	---	---

- $V(10.629,9.932,3)$ is not immediately available, hence we compute it via linear interpolation of $V(10.629,9.700,3)$ and $V(10.629,10,3)$:
  $0.929+(0.929-0.629)/(9.700-10)*(9.932-9.700)=0.697$
- $V(9.409,9.627,3)$ is not immediately available, hence we compute it via linear interpolation of $V(9.409,9.409,3)$ and $V(9.409,9.700,3)$:
  $0+(0-0)/(9.409-9.700)*(9.627-9.409)=0$
- Finally, replacing in the original formula:
  $V(10,9.662,2)=0.998*[0.383*0.697+0.617*0]=0.266$

Asian Strike Option Example

- The entire tree valuation is:

Derivative value at n=3

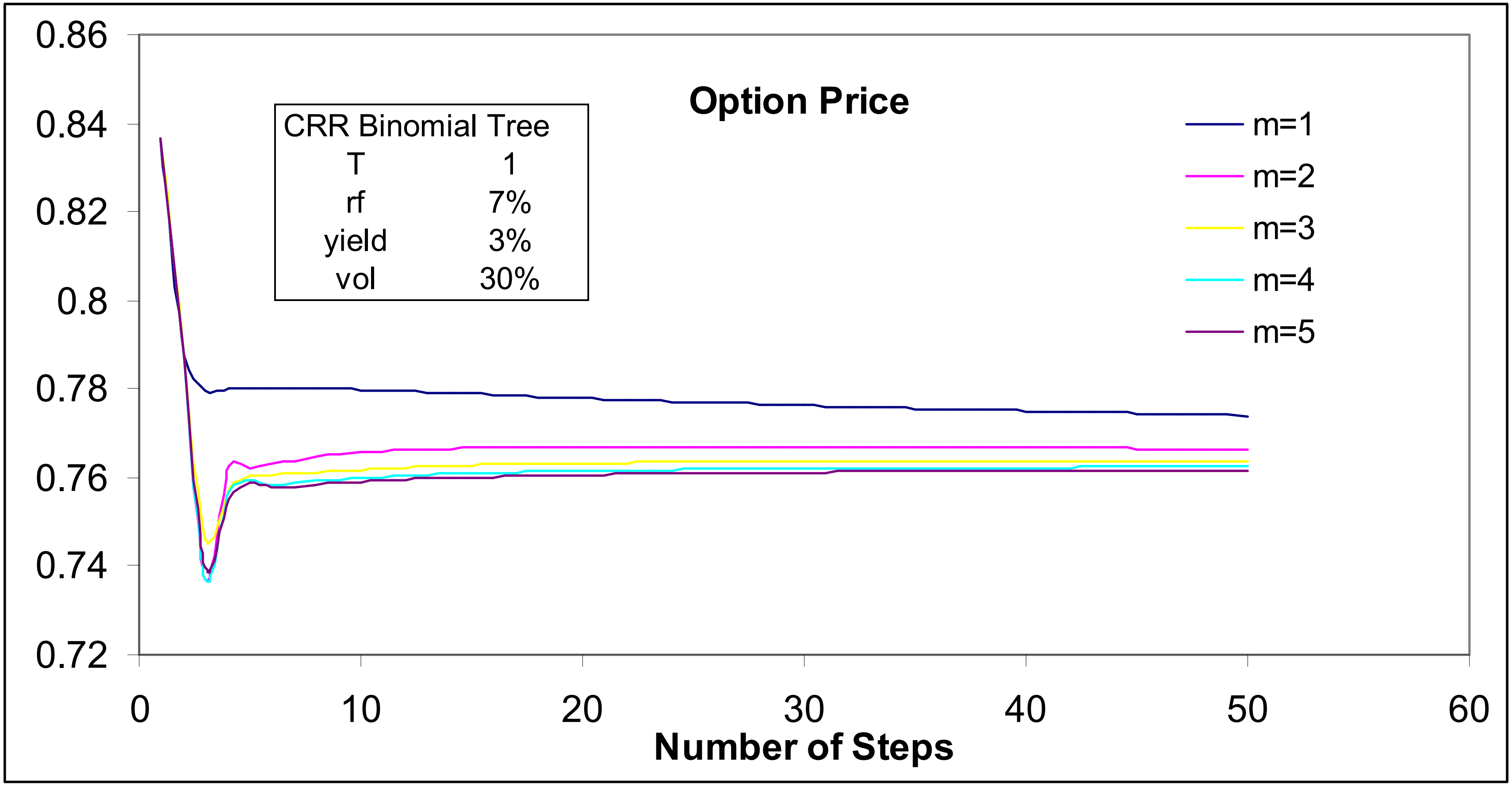
		k	**-3**	**-2**	**-1**	**0**	**1**	**2**	**3**	**4**
min K	**max K**	**price \ Z**	9.126	9.409	9.700	10.000	10.310	10.629	10.958	11.297
3	4	12.007	---	---	---	---	---	---	1.049	0.710
-1	3	10.629	---	---	0.929	0.629	0.319	0.000	0.000	---
-2	2	9.409	---	0.000	0.000	0.000	0.000	0.000	---	---
-3	-2	8.329	0.000	0.000	---	---	---	---	---	---
Derivative value at n=2										
2	3	11.297	---	---	---	---	---	0.395	0.301	---
-1	2	10.000	---	---	0.266	0.180	0.091	0.000	---	---
-2	-1	8.852	---	0.000	0.000	---	---	---	---	---
Derivative value at n=1										
1	2	10.629	---	---	---	---	0.224	0.163	---	---
-1	0	9.409	---	---	0.091	0.069	---	---	---	---
Derivative value at n=0										
0	0	10.000	---	---	---	0.141	---	---	---	---

- Note that we skipped computations wherever not necessary!

Asian Strike Option Example

- We assumed that the average starts fixing today, but what if we were already in the middle of the fixing period?
- We could keep that into account just by setting the value of $A_{0,0}$ to the value accumulated so far, instead of the spot price

Asian Strike Option Example



- Does this chart makes sense?

Asian Strike Option Example

- Note that we made quite a mess: if we price with N steps, we are assuming that we are computing an average with N+1 fixings. So, by increasing the number of steps we are actually changing the payoff!
- To have the correct payoff, we should have a number of time steps identical to the number of fixings
- Or, the number of steps could be a multiple of the number of fixings, but then we would need to consider that for those time steps where there is no fixing the value of the average does not change

Asian Strike Option Example

- In general, the algorithm becomes quite complicated
- Path dependent payoff are not the most suitable for trees! As we shall see, other framework are simpler to implement

Tree Summary

- Strength
  - Handle easily American and Bermuda exercise style
  - Can deal with compounded options, although complicate to handle
- Weaknesses
  - Suffers of curse of dimensionality
  - Need lot of attention in setting up properly the grid for the problem at hand
  - Complicate to deal with path dependencies

Literature Review

- Trees have been a very active area of research in the last 30 years.
- Research focuses on various directions
  - Grid modification to improve convergence speed
  - Analysis of convergence speed
  - Extension to multiple sources of risk (multi-dimensional trees)
  - Extension to path dependent options
  - Extension to new models (e.g. mean reversion, stochastic interest rates)
  - Time varying parameters

Further Readings

- Hull, *Option Future and Other Derivatives*, *Introduction to Binomial Trees*
- San-Lin Chung, *Binomial Model*, Google for : “binomial model review chung” (a brief review of major literature results)
- Figlewski and Gao (1999). *The adaptive mesh model: a new approach to efficient option pricing*, JFE 53
- Boyle (1986), *Option Valuation Using a Three-Jump Process*, International Options Journal 3, 7-12
- Hull J, and A White, 1994, *Numerical Procedures For Implementing Term Structure Models I: Single-Factor Models*, The Journal of Derivatives, Fall, pp. 7-16.
- Bachus, Wu, Zin, *Technical Note on Hull White*
- Chen*, Barrier Option Pricing,* *http://n.ethz.ch/student/rebibr/projects/BarrierOptionPricing.pdf*