Mathematical & Computational Finance I
Lecture Notes

Implied Volatility Trees

April 7 2016
Last update: December 4, 2017

1 Implied Volatility Trees

In the previous set of notes we discussed the Black-Scholes formula and how the binomial model converges to it as we let the number of periods $N \to \infty$. We also introduced the notion of the Black-Scholes implied volatility σ^{BS} that solves the Black-Scholes formula for observed parameters S = market price, K = strike price, r = risk-free interest rate, and τ = time to expiry, and the observed market price of the option V^{obs}.That is, the implied volatility of an option, σ^{BS} solves

$$V(S, K, r, \sigma^{BS}, \tau) = V^{obs}(S, K, \tau)$$

In computing the Black-Scholes implied volatility we typically find that assets generate a volatility "smile" across strike prices/moneyness (usually we find more of a "smirk" for equity and a "smile" for commodities). This informs us that the Black-Scholes model does not represent reality since the model assumes constant volatility across strikes. Additionally, we see non-constant volatility across varying time to contract expiry, again violating the assumption of constant volatility in the Black-Scholes model.

Although the model is clearly lacking, we may still use it since it gives us a market consistent (and easy to compute) way of generating option prices. We use these implied volatilities given by observed options to generate a volatility surface so that we may price derivatives that do not appear in the market (either calls/puts with different strikes/maturities or different structure derivatives altogether).

However, for path-dependent options we are not able to use a simple constant volatility rate. The question emerges of how to price such securities in a market consistent manner? Our solution will be to generate **implied volatility trees**: A volatility process that we fit from observed volatilities and use to price more interesting derivatives. That is, we calibrate our binomial volatility tree to the implied volatility surface and use this model for our needs.

Let us label each node of the tree by (n, j) where n corresponds to the depth in the tree and j corresponds to the number of heads at a given node. Then

$$\begin{aligned} n &= 0, 1, ..., N \quad \text{(time variable)} \\ j &= 0, 1, ..., n \quad \text{(state variable)} \end{aligned}$$

and define

$$\begin{aligned} u(n, j) &= \frac{S(n+1, j+1)}{S(n, j)} \\ d(n, j) &= \frac{S(n+1, j)}{S(n, j)} \\ \tilde{p}(n, j) &= \frac{(1 + r(n, j) - d(n, j)}{u(n, j) - d(n, j)} \end{aligned}$$

to be our up factor, down factor, and risk-neutral transition probability of an up step, respectively.

1.1 Arrow-Debreu Securities

Let $\lambda(n, j)$ be the time-zero value of a derivative security that pays \$1 at node (n, j) and 0 elsewhere. That is, the holder only receives \$1 if the asset price ever reaches (n, j) but nothing in any other scenario. We call such a derivative an Arrow-Debreu security and $\lambda(n, j)$ is the Arrow-Debreu security price. Alternatively, we may also refer to the quantity $\lambda(n, j)$ the state price for a given (n, j), as we did earlier in the course.

We can show that the Arrow-Debreu prices follow a recursive relationship called *Jamshidian's forward induction formula.* We consider a simplified version of Jamshidian's formula presented in the text.

In any version of the binomial model we have the backwards-induction pricing formula

$$V(n, j) = \frac{1}{R(n, j)} \Big[\tilde{p}(n, j)V(n+1, j+1) + \big(1 - \tilde{p}(n, j)\big)V(n+1, j)\Big]$$

for $R(n, j) = 1 + r(n, j)$. We also have that the time-zero price of an arbitrary derivative security can be expressed as

$$V(0, 0) = \sum_{j=0}^{N} \lambda(N, j)V(N, j)$$

That is, we express the time-zero price as a portfolio of Arrow-Debreu securities, weighted by the time N payoffs of the security $\{V(N, 0), ..., V(N, N)\}$.

1.2 Jamshidian's Forward Induction Formula

Set the time-zero Arrow-Debreu security price

$$\lambda(0,0) = 1$$

and set $\lambda(n,j) = 0$ whenever $j < 0$ or $j > n$ (to avoid going out of bounds in the binomial tree). The price at time zero of the Arrow-Debreu security paying \$1 at node (n,j) should be identical to the security with the following time $n-1$ payoff[1]

$$V(n-1,k) = \begin{cases} 0 & \text{if } k > j \\ \frac{1-\tilde{p}(n-1,j)}{R(n-1,j)} & \text{if } k = j \\ \frac{\tilde{p}(n-1,j-1)}{R(n-1,j-1)} & \text{if } k = j-1 \\ 0 & \text{if } k < j-1 \end{cases}$$

for $1 \leq j \leq n-1$. Therefore, for this $1 \leq j \leq n-1$ we may express $\lambda(n,j)$, the time-zero price of the Arrow-Debreu security paying \$1 if we reach node (n,j), by[2]

$$\lambda(n,j) = \frac{1-\tilde{p}(n-1,j)}{R(n-1,j)}\lambda(n-1,j) + \frac{\tilde{p}(n-1,j-1)}{R(n-1,j-1)}\lambda(n-1,j-1)$$

At the boundary $j = 0$ we have the payoff

$$V(n-1,k) = \begin{cases} 0 & \text{if } k > 0 \\ \frac{1-\tilde{p}(n-1,0)}{R(n-1,0)} & \text{if } k = 0 \end{cases}$$

and for $j = n$

$$V(n-1,k) = \begin{cases} \frac{\tilde{p}(n-1,n-1)}{R(n-1,n-1)} & \text{if } k = n-1 \\ 0 & \text{if } k < n \end{cases}$$

Therefore, we express $\lambda(n,j)$ generally by

$$\lambda(n,j) = \begin{cases} \frac{1-\tilde{p}(n-1,j)}{R(n-1,j)}\lambda(n-1,j) & \text{if } j = 0 \\ \frac{1-\tilde{p}(n-1,j)}{R(n-1,j)}\lambda(n-1,j) + \frac{\tilde{p}(n-1,j-1)}{R(n-1,j-1)}\lambda(n-1,j-1) & \text{if } 1 \leq j < n \\ \frac{\tilde{p}(n-1,j-1)}{R(n-1,j-1)}\lambda(n-1,j-1) & \text{if } j = n \end{cases}$$

That is, we have $\lambda(n,j)$ as the time-zero price of the Arrow-Debreu security paying \$1 at node (n,j) given precisely j heads at time n.

[1] This is essentially just the discounted risk-neutral expectation

[2] Essentially, this is a discounted weighted average of how we may reach (n,j) from its preceding nodes.

1.3 Calibration of the Implied Volatility Tree

At time $t = n - 1$ suppose that we have the following data input for our volatility model:

(1) $S(n-1,j)$ for $j = 0, 1, ..., n-1$ (price of the underlying asset for all possible states at time $n-1$).

(2) $V^{put}(n-1,j)$ is the value of a put at node $(n-1,j)$ expiring at time n with strike price $K = S(n-1,j)$.

(3) $V^{call}(n-1,j)$ is the value of a call at node $(n-1,j)$ expiring at time n with strike price $K = S(n-1,j)$.

(4) $R(n-1,j) = 1 + r(n,j)$ for $j = 0, 1, ..., n-1$ given by some interest rate model (i.e. BDT, Ho-Lee, may be state dependent).

(5) $\lambda(n-1,j)$ for $j = 0, 1, ..., n-1$ (Arrow-Debreu prices for possible states at time $n-1$).

We can then use this input to compute $S(n,j)$ for $j = 0, 1, ..., n$. To do so we must consider three possible cases.

1.3.1 Calibration of the Implied Volatility Tree: Case 1

In this case we are at node $(n-1,j)$ (suppose all preceding nodes are successfully calibrated) and that we already know the price given by $S(n,j+1)$ (we have already computed it/it is given to use) and we must calculate $S(n,j)$. That is, we around proceeding downwards in decreasing j for a given time step $n-1$. Our case looks like

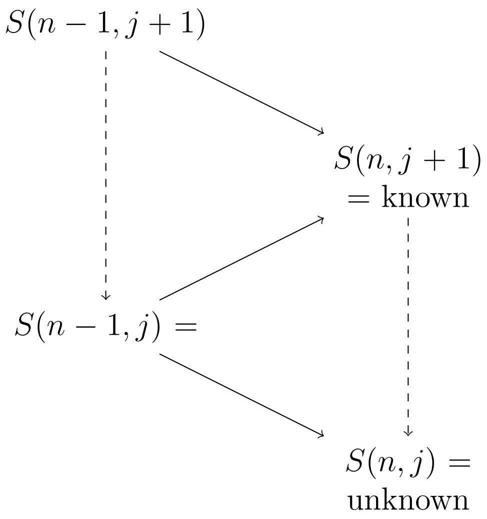

Figure 1: The previous step we had been working at node $(n-1,j+1)$, having computed $S(n,j+2)$ and $S(n,j+1)$. We are presently at node $(n-1,j)$ and wish to compute $S(n,j)$.

Let strike $K = S(n-1,j)$. Then we find the price of a put written on S discounted back to $(n-1,j)$ (risk-neutral pricing)

$$\begin{aligned}
V^{put}(n-1,j) &= \frac{1}{R(n-1,j)}\Big[\tilde{p}(n-1,j)\big[K - S(n,j+1)\big]^+ + \big[1-\tilde{p}(n-1,j)\big]\big[K - S(n,j)\big]^+\Big] \\
&= \frac{1}{R(n-1,j)}\Big[\tilde{p}(n-1,j)\big[0\big] + \big[1-\tilde{p}(n-1,j)\big]\big[K - S(n,j)\big]\Big] \\
&= \frac{1}{R(n-1,j)}\big[1-\tilde{p}(n-1,j)\big]\big[K - S(n,j)\big]
\end{aligned}$$

Recall that we had defined

$$\begin{aligned}
u(n-1,j) &= \frac{S(n,j+1)}{S(n-1,j)} \\
d(n-1,j) &= \frac{S(n,j)}{S(n-1,j)} \\
\tilde{p}(n-1,j) &= \frac{\big(1+r(n-1,j)\big) - d(n-1,j)}{u(n-1,j) - d(n-1,j)} \\
&= \frac{R(n-1,j) - d(n-1,j)}{u(n-1,j) - d(n-1,j)}
\end{aligned}$$

Hence

$$\begin{aligned}
1-\tilde{p}(n-1,j) &= 1 - \frac{\big(1+r(n-1,j)\big) - d(n-1,j)}{u(n-1,j) - d(n-1,j)} \\
&= 1 - \frac{R(n-1,j) - \frac{S(n,j)}{S(n-1,j)}}{\frac{S(n,j+1)}{S(n-1,j)} - \frac{S(n,j)}{S(n-1,j)}} \\
&= 1 - \frac{R(n-1,j) - \frac{S(n,j)}{S(n-1,j)}}{\frac{S(n,j+1)-S(n,j)}{S(n-1,j)}} \\
&= 1 - \frac{S(n-1,j)R(n-1,j) - S(n,j)}{S(n,j+1) - S(n,j)} \\
&= \frac{S(n,j+1) - S(n,j)}{S(n,j+1) - S(n,j)} - \frac{S(n-1,j)R(n-1,j) - S(n,j)}{S(n,j+1) - S(n,j)} \\
&= \frac{S(n,j+1) - S(n,j) - S(n-1,j)R(n-1,j) + S(n,j)}{S(n,j+1) - S(n,j)} \\
&= \frac{S(n,j+1) - S(n-1,j)R(n-1,j)}{S(n,j+1) - S(n,j)} \\
&= \frac{S(n,j+1) - K \cdot R(n-1,j)}{S(n,j+1) - S(n,j)}
\end{aligned}$$

Therefore, substituting this value for $1-\tilde{p}(n-1,j)$ into V^{put} above gives

$$\begin{aligned}
V^{put}(n-1,j) &= \frac{1}{R(n-1,j)}\big[1-\tilde{p}(n-1,j)\big]\big[K-S(n,j)\big] \\
&= \frac{1}{R(n-1,j)}\cdot\frac{S(n,j+1)-K\cdot R(n-1,j)}{S(n,j+1)-S(n,j)}\cdot\big[K-S(n,j)\big]
\end{aligned}$$

and we may solve for the unknown quantity $S(n,j)$ to yield *(steps omitted)*

$$\begin{aligned}
S(n,j) &= \frac{V^{put}(n-1,j)S(n,j+1)+K\left(K-\frac{S(n,j+1)}{R(n-1,j)}\right)}{V^{put}(n-1,j)+K-\frac{S(n,j+1)}{R(n-1,j)}} \\
&= \frac{V^{put}(n-1,j)S(n,j+1)+S(n-1,j)\left(S(n-1,j)-\frac{S(n,j+1)}{R(n-1,j)}\right)}{V^{put}(n-1,j)+S(n-1,j)-\frac{S(n,j+1)}{R(n-1,j)}}
\end{aligned}$$

as desired.[3]

1.3.2 Calibration of the Implied Volatility Tree: Case 2

In this case we are again at node $(n-1,j)$ but we now know the value for $S(n,j)$ and wish to compute $S(n,j+1)$. That is, we around proceeding upwards in increasing j for a given time step $n-1$. Our case looks like

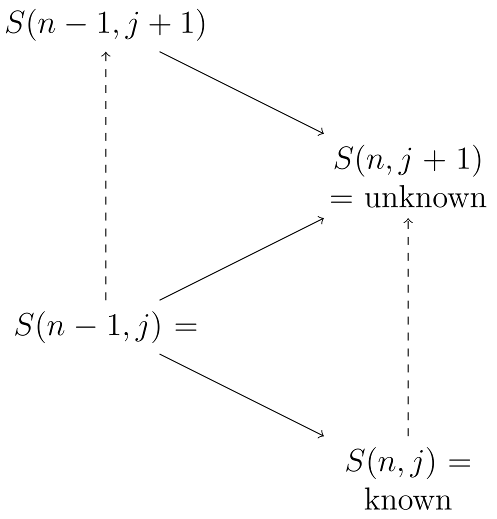

Figure 2: The previous step we had been working at node $(n-1,j)$, having computed $S(n,j-1)$ and $S(n,j)$. We are presently at node $(n-1,j)$ and wish to compute $S(n,j+1)$.

Let $K=S(n-1,j)$ and consider the value of a call option written on S. By the same process as Case 1 we find

$$V^{call}(n-1,j)=\frac{1}{R(n-1,j)}\tilde{p}(n-1,j)[S(n,j+1)-K]$$

[3] Presumably this is more transparent when doing a numerical example.

and we find $\tilde{p}(n-1,j)$ to be written as (using our work above)

$$\tilde{p}(n-1,j) = \frac{S(n-1,j)R(n-1,j) - S(n,j)}{S(n,j+1) - S(n,j)}$$

$$\frac{K \cdot R(n-1,j) - S(n,j)}{S(n,j+1) - S(n,j)}$$

and we may solving for $S(n,j+1)$ to find

$$\begin{aligned} S(n,j+1) &= \frac{V^{call}(n-1,j)S(n,j) + K\left(\frac{S(n,j)}{R(n-1,j)} - K\right)}{V^{call}(n-1,j) + \frac{S(n,j)}{R(n-1,j)} - K} \\ &= \frac{V^{call}(n-1,j)S(n,j) + S(n-1,j)\left(\frac{S(n,j)}{R(n-1,j)} - S(n-1,j)\right)}{V^{call}(n-1,j) + \frac{S(n,j)}{R(n-1,j)} - S(n-1,j)} \end{aligned}$$

as desired.

1.3.3 Calibration of the Implied Volatility Tree: Case 3

In our final case we are at node $(n-1,j)$ but we know neither $S(n,j)$ or $S(n,j+1)$. That is, we wish to price the first (topmost) node when proceeding downwards through the tree at a given time $n-1$ or we wish to price the first (bottommost) node when pricing upwards through the tree.

Let $K = S(n-1,j)$. There are various ways to solve this problem. Our particular approach will assume that

$$\begin{aligned} S(n,j+1) &= K \cdot u(n-1,j) \\ S(n,j) &= K \cdot d(n-1,j) \\ u(n-1,j)d(n-1,j) = 1 \implies d(n-1,j) &= \frac{1}{u(n-1,j)} \end{aligned}$$

We have the fair price for the value of a put option V^{put} at time node $(n-1,j)$ is

$$\begin{aligned}
V^{put} &= \frac{1}{R(n-1,j)}[1-\tilde{p}(n-1,j)][K-S(n,j)] \\
&= \frac{1}{R(n-1,j)}\left[\frac{u(n-1,j)-R(n-1,j)}{u(n-1,j)-d(n-1,j)}\right][K-S(n,j)] \\
&= \frac{1}{R(n-1,j)}\left[\frac{u(n-1,j)-R(n-1,j)}{u(n-1,j)-d(n-1,j)}\right][K-d(n-1,j)S(n-1,j)] \\
&= \frac{1}{R(n-1,j)}\left[\frac{u(n-1,j)-R(n-1,j)}{u(n-1,j)-d(n-1,j)}\right][S(n-1,j)-d(n-1,j)S(n-1,j)] \\
&= \frac{1}{R(n-1,j)}\left[\frac{u(n-1,j)-R(n-1,j)}{u(n-1,j)-d(n-1,j)}\right][1-d(n-1,j)]S(n-1,j) \\
&= \frac{1}{R(n-1,j)}\left[\frac{u(n-1,j)-R(n-1,j)}{u(n-1,j)-d(n-1,j)}\right][1-d(n-1,j)]K \\
&= \frac{1}{R(n-1,j)}\left[\frac{u(n-1,j)-R(n-1,j)}{u(n-1,j)-\frac{1}{u(n-1,j)}}\right]\left[1-\frac{1}{u(n-1,j)}\right]K \\
&= \frac{1}{R(n-1,j)}\left[\frac{u(n-1,j)-R(n-1,j)}{u(n-1,j)-\frac{1}{u(n-1,j)}}\right]\frac{u(n-1,j)-1}{u(n-1,j)}\cdot K \\
&\vdots \qquad\qquad\qquad\qquad\qquad\qquad (\textit{apply somw basic algebra}) \\
&= \frac{1}{R(n-1,j)}\left[\frac{u(n-1,j)-R(n-1,j)}{u(n-1,j)+1}\right]K
\end{aligned}$$

which can be solve for the up factor $u(n-1,j)$ yielding

$$u(n-1,j) = \frac{S(n-1,j)+V^{put}(n-1,j)}{\frac{S(n-1,j)}{R(n-1,j)}-V^{put}(n-1,j)}$$

Then, using this value for $u(n-1,j)$, we recall our earlier modelling specification and compute

$$\begin{aligned}
S(n,j+1) &= K\cdot u(n-1,j) = S(n-1,j)u(n-1,j) \\
S(n,j) &= K\cdot d(n-1,j) = \frac{S(n-1,j)}{u(n-1,j)}
\end{aligned}$$

1.3.4 Calibration of the Implied Volatility Tree: Inputs

For all three cases above we had used the inputs of V^{put} and V^{call}. These values must be calculate from market consistent prices for node $(n-1,j)$ observed at time zero.

Let $P^{(n,K)}(0,0)$ denote the market consistent time-zero price of a European put option with strike K expiring at time n. Likewise, let $C^{(n,K)}(0,0)$ denote the market consistent time-zero price of a European call option with strike K expiring at time n. With these inputs we must calculate both $V^{put}(n-1,j)$ and $V^{call}(n-1,j)$ (with strike $K = S(n-1,j)$, depending on state j) for use in our three cases above. To do so we use an application of risk-neutral pricing using Arrow-Debreu state prices.
For the European put option we find

$$\begin{aligned} P^{(n,K)}(n-1,k) &= \frac{\tilde{p}(n-1,k)(K-S(n,k+1))^+ + (1-\tilde{p}(n-1,k))(K-S(n,k))^+}{R(n-1,k)} \\ &= \frac{0 + (1-\tilde{p}(n-1,k))(K-S(n,k))}{R(n-1,k)} \\ &= \frac{1}{R(n-1,k)}(1-\tilde{p}(n-1,k))(K-S(n,k)) \\ &= \frac{1}{R(n-1,k)}[K-S(n-1,k)R(n-1,k)] \quad \text{(I don't see this step...)} \end{aligned}$$

where the right hand side is easily computable with known quantities when at node $(n-1,k)$.

We have the requirements

$$\begin{aligned} P^{(n,K)}(n-1,k) &= 0 \quad \text{if } k > j \\ P^{(n,K)}(n-1,j) &= V^{put}(n-1,j) \quad \text{else} \end{aligned}$$

where $P^{(n,K)}$ is the market consistent price of a put expiring at time n with strike K and V^{put} corresponds to the price generated by our model.[4] We may express our put in terms of Arrow-Debreu prices[5]

$$\begin{aligned} P^{(n,K)}(0,0) &= \sum_{k=0}^{j} \lambda(n-1,k)P^{(n,K)}(n-1,k) \\ &= \sum_{k=0}^{j-1} \lambda(n-1,k)P^{(n,K)}(n-1,k) + \lambda(n-1,j)V^{put}(n-1,j) \\ &= \Sigma_P(n-1,j) + \lambda(n-1,j)V^{put}(n-1,j) \end{aligned}$$

where

$$\Sigma_P(n-1,j) := \sum_{k=0}^{j-1} \lambda(n-1,k)P^{(n,K)}(n-1,k)$$

Therefore,

$$\begin{aligned} P^{(n,K)}(0,0) &= \Sigma_P(n-1,j) + \lambda(n-1,j)V^{put}(n-1,j) \\ \implies V^{put}(n-1,j) &= \frac{P^{(n,K)}(0,0) - \Sigma_P(n-1,j)}{\lambda(n-1,j)} \end{aligned}$$

[4] There's something I don't understand about the difference between the two.

[5] We only sum over the nodes where the payoff is > 0, this is why we end at $k = j$.

for $K = S(n-1, j)$. Likewise, for $V^{call}(n-1, j)$ we find

$$V^{call} = \frac{P^{(n,K)}(0,0) - \Sigma_C(n-1,j)}{\lambda(n-1,j)}$$

where $K = S(n-1, j)$ and[6]

$$\Sigma_C(n-1,j) := \sum_{k=j+1}^{n-1} \lambda(n-1,k) C^{(n,K)}(n-1,k)$$

At each stage in the procedure we must test the condition/spacing rule

$$S(n,j) < S(n-1,j)R(n-1,j) < S(n,j+1)$$

otherwise our model would permit arbitrage. It should be clear to us that this is similar to our earlier requirement of $d < 1 + r < u \implies dS_n < (1+r)S_n < uS_n$. The reason we have this requirement is because the forward price of the asset at node $(n-1, j)$ for delivery at n is given by[7]

$$F(n-1,j) = S(n-1,j)R(n-1,j)$$

If this condition is not satisfied then we may encounter $\tilde{p}(n-1,j) > 1$ or $\tilde{p}(n-1,j) < 0$. In either case we can show that we an arbitrage portfolio involving the forward contract exists.

But what happens if the spacing rule $S(n,j) < S(n-1,j)R(n-1,j) < S(n,j+1)$ is violated? In this case the option price that is produce is discarded and we select a stock price that maintains the (log)[8] spacing between the node and adjacent nodes.

(1) If $S(n, j+1)$ is known and we have $S(n,j) > S(n-1,j)R(n-1,j)$ then we replace $S(n, j)$ by the procedure

$$S'(n,j) = \frac{S(n,j+1)S(n-1,j-1)}{S(n-1,j)}$$

(2) If $S(n, j)$ is known and $S(n,j+1) < S(n-1,j)R(n-1,j)$ we replace $S(n, j+1)$ by the procedure

$$S'(n,j+1) = \frac{S(n,j)S(n-1,j-1)}{S(n-1,j)}$$

Furthermore, we must also not use the values $P^{(n,K=S(n-1,j))}$ and $C^{(n,K=S(n-1,j))}$ which had produced this situation.

[6] We only sum over the nodes where the payoff is > 0, this is why we start at $k = j + 1$.

[7] This is something that should be verified...

[8] Why log?

Example: Consider a $N = 4$ step implied volatility tree ($n = 0, 1, 2, 3, 4$) with inputs $S(0, 0) = 90$, $r = 0.05$ (continuously compounded), $R(n, j) = e^r$ for all n, $\Delta t = 1$, and the time-dependent implied volatility surface function (smile) is given by

$$\sigma_{imp} = 0.15 + 0.1 \left(1 - \frac{K}{90}\right)^2$$

Solution:

At time $\boldsymbol{n = 0}$ we are given that $S(0, 0) = 90$. We know neither the of the following stock prices $S(1, 0)$ or $S(0, 1$ and so we see that we are in Case 3.

For this case we require $V^{put}(0, 0)$, the time-zero price of a put expiring at time 1 with strike $K = S(0, 0)$. Denote $P^{(1,90)}(0, 0)$ to be the Black-Scholes price at time zero of such a put with implied volatility $\sigma_{imp}(90) = 0.15$. Using the Black-Scholes formula with $r = 0.05$ we find

$$P^{(1,90)}(0, 0) = 3.343141$$

Recall that we had managed to express the model price $V^{put}(n - 1, j)$ as

$$\begin{aligned}
P^{(n,K)}(0, 0) &= \sum_{k=0}^{j} \lambda(n - 1, k) P^{(n,K)}(n - 1, k) \\
&= \sum_{k=0}^{j-1} \lambda(n - 1, k) P^{(n,K)}(n - 1, k) + \lambda(n - 1, j) V^{put}(n - 1, j) \\
\implies V^{put}(n - 1, j) &= \frac{P^{(n,K)}(0, 0) - \sum_{k=0}^{j-1} \lambda(n - 1, k) P^{(n,K)}(n - 1, k)}{\lambda(n - 1, j)} \\
\implies V^{put}(0, 0) &= \frac{P^{(1,K)}(0, 0) - 0}{\lambda(0, 0)}
\end{aligned}$$

Since the Arrow-Debreu security price given by $\lambda(0, 0) = 1$ we find

$$V^{put}(0, 0) = \frac{P^{(1,90)}(0, 0)}{\lambda(0, 0)} = 3.343141$$

Now, to compute our up and down factors we use our formula above

$$\begin{aligned}
u(n - 1, j) &= \frac{S(n - 1, j) + V^{put}(n - 1, j)}{\frac{S(n-1,j)}{R(n-1,j)} - V^{put}(n - 1, j)} \\
\implies u(0, 0) &= \frac{S(0, 0) + V^{put}(0, 0)}{\frac{S(0,0)}{R(0,0)} - V^{put}(0, 0)} \\
&= \frac{90 + 3.343131}{\frac{90}{e^{0.05}} - 3.343131} \\
&= 1.134629 \\
\implies d(0, 0) &= \frac{1}{u(0, 0)} = 0.881345
\end{aligned}$$

Thus

$$
\begin{aligned}
S(1,1) &= u(0,0)S(0,0) = 1.134629 \cdot 90 = 102.1167 \\
S(1,0) &= d(0,0)S(0,0) = 0.881345 \cdot 90 = 79.32105
\end{aligned}
$$

and note that the condition $S(1,0) < R(0,0)S(0,0) < S(1,1)$ is satisfied:

$$79.32105 < 94.6144 < 102.1167$$

so we have no adjustments to make with respect to $S(1,0)$ and $S(1,1)$.

Before moving on to time $n = 1$ we calculate the risk-neutral probability $\tilde{p}(0,0)$ in the typical way

$$
\begin{aligned}
\tilde{p}(0,0) &= \frac{R(0,0) - d(0,0)}{u(0,0) - d(0,0)} \\
&= \frac{e^{0.05} - 0.881345}{1.134629 - 0.881345} = 0.6708903
\end{aligned}
$$

as well as the corresponding Arrow-Debreu state prices for nodes $(1,0)$ and $(1,1)$ using Jamshidian's formula

$$
\begin{aligned}
\lambda(1,1) &= \frac{\tilde{p}(0,0)}{R(0,0)}\lambda(0,0) = \frac{0.6708903}{e^{0.05}} = 0.6381706 \\
\lambda(1,0) &= \frac{1 - \tilde{p}(0,0)}{R(0,0)}\lambda(0,0) = \frac{1 - 0.6708903}{e^{0}.05} = 0.3130589
\end{aligned}
$$

At time $n = 1$ we now know both $S(1,1)$ and $S(1,0)$ from the previous step and seek to compute $S(2,0), S(2,1), S(2,2)$. Since our process is symmetric (by $d = \frac{1}{u}$) we may set $S(2,1) = 90$ and so we may avoid Case 3. We find that we are at

(i) Case 1: We know $S(2,1)$ and seek $S(2,0)$.

(ii) Case 2: We know $S(2,1)$ and seek $S(2,2)$.

We are presently at node $(1,0)$. We let $K = S(1,0) = 79.32105$. In order to find the subsequent stock prices $S(2,0)$ and $S(2,2)$ we must first find $V^{put}(1,0)$, the node $(1,0)$ value of a put expiring at time 2 with strike K.

Let $P^{(2,79.32105)}(0,0)$ be the time-zero Black-Scholes price of a put expiring at time 2 with strike $K = 79.32105$ and implied volatility

$$\sigma_{imp}(79.32105) = 0.15 + 0.1\left(1 - \frac{79.32105}{90}\right)^2 = 0.1514079$$

We can use the Black-Scholes formula with parameters $S_0 = 90, K = 79.32105, T = 2, t = 0, r = 0.05, \sigma = 0.1514079$ to find

$$P^{(2,79.32105)}(0,0) = 1.280316$$

Therefore, again using our formula for V^{put} with Arrow-Debreu securities

$$\begin{aligned} P^{(n,K)}(0,0) &= \sum_{k=0}^{j-1} \lambda(n-1,k)P^{(n,K)}(n-1,k) + \lambda(n-1,j)V^{put}(n-1,j) \\ \implies V^{put}(1,0) &= \frac{P^{(2,K)}(0,0) - 0}{\lambda(1,0)} \\ &= \frac{1.280316}{0.3130589} = 4.089698 \end{aligned}$$

Then, we price $S(2,0)$ from Case 1 (since we know the "above" value $S(2,1)$)

$$\begin{aligned} S(2,0) &= \frac{V^{put}(1,0)S(2,1) + S(1,0)[S(1,0) - \frac{S(2,1)}{R(1,0)}]}{V^{put}(1,0) + S(1,0) - \frac{S(2,1)}{R(1,0)}} \\ &= \frac{4.089698 \cdot 90 + 79.32105[79.32105 - \frac{90}{e^{0}.05}}{4.089698 + 79.32105 - \frac{90}{e^{0}.05}} = 59.46849 \end{aligned}$$

Checking our no-arbitrage condition $S(2,0) < R(1,0)S(1,0) < S(2,1)$

$$59.46849 < 83.38792 < 90$$

is satisfied.

Now consider node $(1,1)$. We let the strike price be the asset value at this node. That is, let $K = S(1,1) = 102.1167$. Our goal is to price $S(2,2)$ using the known quantities "below" it, and so we find ourselves in Case 2 (we require the value of a call option $V^{call}(1,1)$ to do so).

Let $C^{(2,K)}(0,0)$ be the Black-Scholes price at time zero of a call option expiring at time 2 with strike $K = 102.1167$ and implied volatility

$$\sigma_{imp}(102.1167) = 0.15 + 0.1\left(1 - \frac{102.1167}{90}\right)^2 = 0.1518125$$

Using the Black-Scholes formula we the appropriate parameters we find

$$C^{(2,K)}(0,0) = 6.655091$$

Again using our formula for V^{call} with Arrow-Debreu securities (with $n = 2, j = 1$)

$$\begin{aligned} C^{(n,K)}(0,0) &= \sum_{k=0}^{j-1} \lambda(n-1,k)C^{(n,K)}(n-1,k) + \lambda(n-1,j)V^{call}(n-1,j) \\ \implies C^{(2,K)}(0,0) &= \lambda(1,0)C^{(2,K)}(1,0) + \lambda(1,1)V^{call}(1,1) \\ &= 0 + \lambda(1,1)V^{call}(1,1) \quad (C^{(2,K)}(1,0) \text{ cannot expire ITM}) \\ \implies V^{call}(1,1) &= \frac{C^{(2,K)}(0,0)}{\lambda(1,1)} \\ &= \frac{6.655091}{0.6381706} = 10.42839 \end{aligned}$$

Thus, applying our result from Case 2

$$\begin{aligned}
S(n,j+1) &= \frac{V^{call}(n-1,j)S(n,j) + S(n-1,j)\left(\frac{S(n,j)}{R(n-1,j)} - S(n-1,j)\right)}{V^{call}(n-1,j) + \frac{S(n,j)}{R(n-1,j)} - S(n-1,j)} \\
\implies S(2,2) &= \frac{V^{call}(1,1)S(2,1) + S(1,1)\left(\frac{S(2,1)}{R(1,1)} - S(1,1)\right)}{V^{call}(1,1) + \frac{S(2,1)}{R(1,1)} - S(1,1)} \\
&= \frac{10.42839 \cdot 90 + 102.1167\left(\frac{90}{e^{0}.05} - 102.1167\right)}{10.42839 + \frac{90}{e^{0}.05} - 102.1167} = 122.9072
\end{aligned}$$

Once again confirming our condition $S(2,1) < R(1,1)S(1,1) < S(2,2)$:

$$90 < 107.3523 < 122.9072$$

is satisfied. Finally, we calculate the risk-neutral probabilities and corresponding Arrow-Debreu prices[9]

$$\begin{aligned}
\tilde{p}(1,0) &= \frac{R(1,0)S(1,0) - S(2,0)}{S(2,1) - S(2,0)} \\
&= \frac{e^{0}.05 \cdot 79.32105 - 59.46849}{90 - 59.46849} = 0.7834344 \\
\tilde{p}(1,1) &= \frac{R(1,1)S(1,1) - S(2,1)}{S(2,2) - S(2,1)} \\
&= \frac{e^{0}.05 \cdot 102.1167 - 90}{122.9072 - 90} = 0.5273092
\end{aligned}$$

and

$$\begin{aligned}
\lambda(2,0) &= \frac{1-\tilde{p}(1,0)}{R(1,0)}\lambda(1,0) = 0.06449126 \\
\lambda(2,1) &= \frac{1-\tilde{p}(1,1)}{R(1,1)}\lambda(1,1) + \frac{\tilde{p}(1,0)}{R(1,0)}\lambda(1,0) = 0.5202449 \\
\lambda(2,2) &= \frac{\tilde{p}(1,1)}{R(1,1)}\lambda(1,1) = 0.3201013
\end{aligned}$$

At time $\boldsymbol{n = 2}$ we have computed $S(2,0), S(2,1)$, and $S(2,2)$.

Consider first the central node $(2,1)$: From here we know neither $S(3,1)$ or $S(3,2)$. Thus, we are in Case 3. Let $K = S(2,1) = 90$. For this case we must find $V^{put}(2,1)$: The value of a put at node $(2,1)$ expiring at time 3 with strike $K = 90$.

Let $P^{(3,K)}(0,0)$ be the Black-Scholes price at time zero of such a put. We can find its implied volatility to be

$$\sigma_{imp}(K = 90) = 0.15$$

[9] We use $\tilde{p} = \frac{(1+r)-d}{u-d} = \frac{(1+r)S-dS}{uS-dS}$.

and so the Black-Scholes price is

$$P^{(3,K)}(0,0) = 3.778657$$

Thus, using the Arrow-Debreu securities we find, with $n = 3, j = 1$,

$$\begin{aligned} P^{(3,K)}(0,0) &= \sum_{k=0}^{j-1} \lambda(n-1,j)P^{(n,K)}(n-1,j) + \lambda(n-1,j)V^{put}(n-1,j) \\ &= \lambda(2,0)P^{(3,K)}(2,0) + \lambda(2,1)V^{put}(2,1) \end{aligned}$$

From the risk-neutral pricing expansion (done earlier) we can write

$$\begin{aligned} \lambda(2,0)P^{(3,K)}(2,0) &= \lambda(2,0)\frac{1}{R(2,0)}\left[K - S(2,0)R(2,0)\right] \\ &= 0.06449126 \cdot 26.14216 = 1.68594 \end{aligned}$$

Hence[10]

$$\begin{aligned} V^{put}(2,1) &= \frac{P^{(3,K)}(0,0) - \lambda(2,0)P^{(3,K)}(2,0)}{\lambda(2,1)} \\ &= \frac{3.778657 - 1.68594}{0.5202449} = 4.022561 \end{aligned}$$

Then, by Case 3, we continue with computing

$$u(2,1) =$$

[10] Why couldn't we just use the risk-neutral pricing expansion directly to price $V^{put}(2,1) = P^{(3,K)}(2,1)$?