用微分數較正利率模型

Calibrating Interest Rate Models with the Differential Tree Method: the Case of the Black-Derman-Toy Model

Wei-Jui Chen (陳偉叡) & Yuh-Dauh Lyuu (呂育道) *

Department of Computer Science & Information Engineering
National Taiwan University
(臺灣大學資訊工程系)

November 5, 1997

Abstract

在一元利率模型中，所有的有價證券及利率皆決定於一個變數，也就是短期利率。市場的殖利率及其變動性則用來建立未來的短期利率樹，這個動作稱爲&梶縧C這樹可以用來對利率衍生物做計價的工作。本文之貢獻在於提出一個有效率校正利率樹的一般性演算法，它用到呂育道*[1995]*所提出的微分樹及*Jamshidian[1991]*所提出的*forward induction*。時間因此可以壓到 $O\left(n^2\right)$ 空間則到 $O(n)$。

In one simple and versatile model of interest rates, all security and rates depend on only one factor—the short rate. The current structure of long rates and their estimated volatilities are used to construct a tree of possible future short rates. This tree can then be used to value interest rate-sensitive securities.

The main contribution of this paper is a general approach to calibrate the tree of short rates efficiently. Two important concepts are introduced that can greatly speed up the process and save much space. They are the concepts of differential tree *due to Lyuu (1995) and* forward induction *due to Jamshidian (1991). With them, the running time can be reduced to* $O\left(n^2\right)$ *and the space to* $O(n)$*, where n is the number of time periods. The results are very encouraging. For example, interest rate trees with hundreds of periods can usually be calibrated within 20 seconds. Even trees with up to 2,000 periods can constructed within 100 seconds.*

1 Motivations

Interest rate-contingent claims such as caps, floors, swaptions, bond options, warrants, captions, and mortgage-backed securities have been popular in recent years. The valuation of these instruments is now a major concern for both practitioners and academics. To achieve that, one needs good methods to describe the evolution of yield curves.

During the past twenty years, there have been many attempts to describe yield curve movements using *one-factor models*. See Hull (1997) for a survey. In 1990, Black, Derman, and Toy used a binomial tree to construct a one-factor model of the short rates that fit the current term structure of all discount bonds. Given a market term structure and the resulting binomial tree of the short rates, a model can be used to value interest rate-sensitive securities. Take the bond option as an example. First the future prices of a Treasury bond at various points in time are found. These prices are used to determine the option's value at expiration. Given the values of a call or put at expiration, their possible values before expiration can be found by the same discounting procedure used to value the bond.

To get accurate solutions for option values, we need a tree that is finely spaced between today and the option's or even the underlying asset's expiration. Ideally, we would like a tree with one-day steps and a 30-year horizon so that coupon payments and option exercise dates would always fall precisely on a node.

In practice, it may cost much memory to build a 30-year tree with daily periods, and it may take hours to value a security. So, we introduce a general approach to calibrate the tree of short rates efficiently. Two important concepts are introduced that can greatly speed up the process. They are concepts of the *differential tree method* and *forward induction*. The space issue is addressed by the specific nature of the model, which allows for very efficient representation.

*Master thesis version is winner of a 1997 Dragon Best Thesis Award, Acer Foundation.

2 Binomial Interest Rate Trees

A *term structure model* is a model that describes the probabilistic behavior of interest rates. They are more complicated than the models used to describe stock price or exchange rate. This is because they are concerned with movements in the entire yield curve, not just with changes in a single variable. As time passes, the individual interest rates in the term structure change. In addition, the shape of the curve itself is liable to change as well.

The goal of this section is to present an *arbitrage-free* discrete-time binomial term structure model which has enough flexibility to explain the different features of interest rate movements and allows the valuation of a rich class of interest rate derivatives. The idea of arbitrage-free model is due to Ho and Lee (1986). The alternatives are the so-called *equilibrium models.*

The binomial tree is constructed so that the logarithm of the short rate, or future one-period spot rate, obeys a binomial distribution. In this way, the limiting distribution for the short rate at any time is lognormal.

In the binomial interest rate process, a binomial tree of possible short rates for each future period is constructed. Each short rate is followed by two short rates as the possible outcomes of a previous time period. In Figure 1, node A is associated with the start of period j *during* which short rate r is in effect. At the conclusion of period j, a new short rate goes into effect for period $j+1$. This may take one of two possible values: r_l, the "low" short rate outcome for period $j+1$ shown at node B, or r_h, the "high" short rate outcome at node C. Each of r_l and r_h has a fifty percent chance of occurring, and they are the only possibilities for period $j+1$'s short rate from node A.

As the binomial process unfolds, we should make sure the paths recombine. The result is that the logarithm of the one-period rate obeys a binomial distribution with $p = 0.5$. In this way, the limiting distribution for the one-period rate becomes lognormal.

Suppose the short rate r can go to r_h and r_l with equal probability in a period of length Δt. Percent volatility of short rate, $\Delta r/r$, is

$$\sigma = \frac{1}{2} \times \frac{1}{\sqrt{\Delta t}} \times \ln(r_h/r_l)$$

when the short rate follows a lognormal process in the limit (Lyuu (1995)). So $(1/2)\ln(r_h/r_l) = \sigma\sqrt{\Delta t}$. Note that, as

$$\frac{r_h}{r_l} = e^{2\sigma\sqrt{\Delta t}}, \qquad (1)$$

greater volatility, hence uncertainty, leads to larger r_h/r_l and wider ranges of possible short rates. The ratio is a constant across time if the volatility is a constant. Note also that r_h/r_l has nothing to do with r yet. To nail down the values of r_h and r_l, we need information from the current yield curve, and it is this information that establishes the relationship between r and its two successors, r_l and r_h. Equation (1) is a fundamental building block for the binomial interest rate tree.

We now proceed beyond the first period. In general, there are j possible rates for period j. According to the binomial process, the rates are

$$r_j, r_j v_j, r_j {v_j}^2, \ldots, r_j {v_j}^{j-1}, \qquad (2)$$

where

$$v_j = e^{-2\sigma_j\sqrt{\Delta t}}$$

is the multiplicative ratio for the rates in period j. We shall call r_j the *baseline rate.* Figure 2 depicts the resulting tree structure. One salient feature of the tree is *path independence*, that is, the term structure at any node is independent of the path taken to reach it.

3 The Black-Derman-Toy (BDT) Model

The model has three key features.

1. Its fundamental variable is the short rate—the annualized one period interest rate. This short rate is the one factor of the model; its change drives all security prices.

2. The model takes as inputs an array of yields on zero-coupon Treasury bonds for various maturities and an array of yield volatilities for the same bonds. We call the first array the *yield curve* and the second the *volatility curve.* Together, these curves form the market *term structure.*

3. The model internally adjusts an array of baseline rates and an array of volatilities for the future spot rate to match the inputs.

We examine how the model works in an imaginary world in which changes in all bond yields are perfectly correlated, expected returns on all securities over one period are equal, short rates at any time are lognormally distributed, and there are no taxes or trading costs.

In the BDT model, all we want to do is to find the short rate tree that can match the market term structure. We will find the rates starting from now and

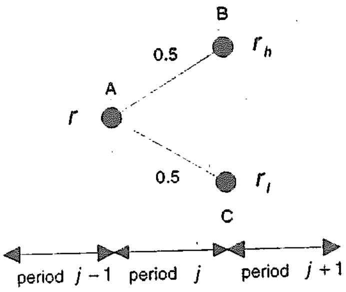

Figure 1: BINOMIAL INTEREST RATE PROCESS. From node A, there are two equally likely scenarios for the short rate: r_l and r_h. Rate r is applicable to node A for period j, rate r_l is applicable to node B for period $j+1$, and rate r_h is applicable to node C for period $j+1$.

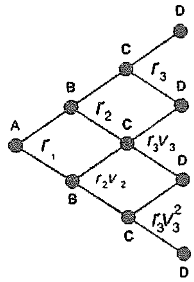

Figure 2: BINOMIAL INTEREST RATE TREE. The distribution at any time converges to a lognormal distribution. The baseline rates are r_1, r_2, r_3, ...

onward into the future. Remember, however, that the BDT model assumes that there are n possible short rates for period n. The n short rates are

$$r_n {v_n}^2, \ldots, r_n {v_n}^{n-1},$$

where r_n is the baseline rate. Suppose the price of the n-period zero-coupon bond moves up to p_1 and down to p_2 one period from now (time zero). Obviously, p_1 and p_2 are functions of r_n and v_n. We can get

$$\frac{\frac{1}{2}p_1 + \frac{1}{2}p_2}{1+r_1} = \frac{1}{(1+y_0(n))^n}, \quad (3)$$

where $y_0(n)$ is the current n-period spot rate, which is known.

Viewed from time zero, the future $(n-1)$-period spot rate at period one are uncertain. Let $y_{11}(n-1)$ represent the $(n-1)$-period spot rate at the upper node one period from now, $y_{10}(n-1)$ represent the $(n-1)$-period spot rate at the lower node one period from now, and $\sigma_0^2(n)$ represent the variance viewed from date zero of the $(n-1)$-period spot rate one period from now. The variance calculation depends on the assumptions made regarding the interest rate process. Since we assume short-term rates are lognormally distributed, the logarithms of the rates follow a normal distribution and the appropriate variance calculation is given by

$$\begin{aligned} \sigma_0^2(n) &= p(1-p)[\ln y_{11}(n-1) - \ln y_{10}(n-1)]^2 \\ &= p(1-p)\left[\ln \frac{y_{11}(n-1)}{y_{10}(n-1)}\right]^2 \end{aligned}$$

Hence, for $p = 1/2$,

$$\sigma_0(n) = (1/2)\ln\left(\frac{y_{11}(n-1)}{y_{10}(n-1)}\right) \quad (4)$$

Recall that the bonds are zero-coupon bonds. So,

$$\begin{aligned} p_1 &= \frac{1}{[1+y_{11}(n-1)]^{n-1}} \\ p_2 &= \frac{1}{[1+y_{10}(n-1)]^{n-1}} \end{aligned}$$

In other words,

$$y_{11}(n-1) = p_1^{\frac{-1}{n-1}} - 1 \quad (5)$$

$$y_{10}(n-1) = p_2^{\frac{-1}{n-1}} - 1 \quad (6)$$

Substitute (5) and (6) into (4) to get

$$\sigma_0(n) = \frac{\ln\left(\frac{p_1^{-\frac{1}{n-1}}-1}{p_2^{-\frac{1}{n-1}}-1}\right)}{2}. \quad (7)$$

Rearranging (3) and (7) in the form of simultaneous equations, we have

$$\begin{aligned} f(p_1,p_2) &\equiv {p_1}^{-\frac{1}{n-1}} - 1 - e^{2\sigma_0(n)}\left({p_2}^{-\frac{1}{n-1}} - 1\right) = 0 \quad (8) \\ g(p_1,p_2) &\equiv p_1 + p_2 - \frac{2(1+r_1)}{[1+y_0(n)]^n} = 0 \quad (9) \end{aligned}$$

The above shows we have to solve two simultaneous nonlinear equations.

4 Calibrating BDT Model with Differential Tree

4.1 Solving systems of nonlinear equations

To solve equations like (8) and (9) simultaneously requires numerical methods. Let (x_k, y_k) be the kth approximation to the solution of two simultaneous equations,

$$f(x,y)=0 \text{ and } g(x,y)=0.$$

The Newton-Raphson method leads to the following linear equations for the $(k+1)$st approximation, (x_{k+1}, y_{k+1}),

$$\begin{bmatrix} \frac{\partial f(x_k,y_k)}{\partial x} & \frac{\partial f(x_k,y_k)}{\partial y} \\ \frac{\partial g(x_k,y_k)}{\partial x} & \frac{\partial g(x_k,y_k)}{\partial y} \end{bmatrix} \begin{bmatrix} \Delta x_{k+1} \\ \Delta y_{k+1} \end{bmatrix} = - \begin{bmatrix} f(x_k,y_k) \\ g(x_k,y_k) \end{bmatrix}$$

where $\Delta x_{k+1} \equiv x_{k+1} - x_k$ and $\Delta y_{k+1} \equiv y_{k+1} - y_k$. The above equations have a unique solution for $(\Delta x_{k+1}, \Delta y_{k+1})$ when the *Jacobian determinant* of f and g,

$$J = \begin{vmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \end{vmatrix}$$

does not vanish at (x_k, y_k). The $(k+1)$st approximation is simply $(x_k + \Delta x_{k+1}, y_k + \Delta y_{k+1})$. See Hildebrand (1974).

4.2 The differential tree method

To solve nonlinear equations like (8) and (9) by the Newton-Raphson method, we need to calculate derivatives. This is accomplished by the differential tree method due to Lyuu (1995). Take Figure 3 as an example to describe the idea of differential tree.

Suppose r such that $P(r) = 0$ is to be calculated with the Newton-Raphson method. Node A is the root of a subtree of the original tree. In the process of computing the price $p_A(r)$ at node A, money at A's two successor nodes, B and C, will be discounted by a factor of $1/(1+r_A)$. Note that r_A is known. We need to compute $p'_A(r)$ as well to use the Newton-Raphson method. Since

$$p_A(r) = c + \frac{p_B(r) + p_C(r)}{2\,(1+r_A)},$$

where c denotes the cash flow at node A,

$$p'_A(r) = \frac{p'_B(r) + p'_C(r)}{2\,(1+r_A)}. \tag{10}$$

Hence, if, by induction, $p'_B(r)$ and $p'_C(r)$ are both available, then computing $p'_A(r)$ is easy. Applying the above argument recursively, we will eventually arrive at the root of the tree with $p(r)$ and $p'(r)$ simultaneously. We shall call it the differential tree method.

We proceed to find the short rate tree by combining the differential tree method and the Newton-Raphson method. Since p_1 and p_2 are functions of r_n and v_n, $f(p_1,p_2)$ and $g(p_1,p_2)$ are also functions of r_n and v_n, say $F(r_n, v_n)$ and $G(r_n, v_n)$. Now we want to solve $F(r_n, v_n) = 0$ and $G(r_n, v_n) = 0$. For the sake of expression, we use $F(r, v)$ instead of $F(r_n, v_n)$ and $G(r, v)$ instead of $G(r_n, v_n)$. As mentioned in Section 4.1, the $(k+1)$st approximation, (r_{k+1}, v_{k+1}), satisfies

$$\begin{bmatrix} \frac{\partial F(r_k,v_k)}{\partial r} & \frac{\partial F(r_k,v_k)}{\partial v} \\ \frac{\partial G(r_k,v_k)}{\partial r} & \frac{\partial G(r_k,v_k)}{\partial v} \end{bmatrix} \begin{bmatrix} \Delta r_{k+1} \\ \Delta v_{k+1} \end{bmatrix} = - \begin{bmatrix} F(r_k,v_k) \\ G(r_k,v_k) \end{bmatrix},$$

where $\Delta r_{k+1} =\equiv r_{k+1} - r_1$, and $\Delta v_{k+1} \equiv v_{k+1} - v_k$. So we need to evaluate $\frac{\partial F}{\partial r}$, $\frac{\partial F}{\partial v}$, $\frac{\partial G}{\partial r}$ and $\frac{\partial G}{\partial v}$. Obviously,

$$\begin{aligned} \frac{\partial G}{\partial r} &= \frac{\partial p_1}{\partial r} + \frac{\partial p_2}{\partial r} \\ \frac{\partial G}{\partial v} &= \frac{\partial p_1}{\partial v} + \frac{\partial p_2}{\partial v} \end{aligned}$$

And by the chain rule,

$$\begin{aligned} \frac{\partial F}{\partial r} &= \frac{\partial f}{\partial p_1}\frac{\partial p_1}{\partial r} + \frac{\partial f}{\partial p_2}\frac{\partial p_2}{\partial r} \\ \frac{\partial F}{\partial v} &= \frac{\partial f}{\partial p_1}\frac{\partial p_1}{\partial v} + \frac{\partial f}{\partial p_2}\frac{\partial p_2}{\partial v} \end{aligned}$$

Take the above four equations into consideration, the common items that we need to evaluate are $\frac{\partial p_1}{\partial r}$, $\frac{\partial p_2}{\partial r}$, $\frac{\partial p_1}{\partial v}$, $\frac{\partial p_2}{\partial v}$, $\frac{\partial f}{\partial p_1}$ and $\frac{\partial f}{\partial p_2}$. Evaluating $\frac{\partial p_1}{\partial r}$, $\frac{\partial p_2}{\partial r}$, $\frac{\partial p_1}{\partial v}$, $\frac{\partial p_2}{\partial v}$ directly is cumbersome. Fortunately, the tree, already a compact representation of $p_1(r, v)$ and $p_2(r, v)$, can be used to compute all of them in one sweep. This is where the differential tree method would come into play. As for $\frac{\partial f}{\partial p_1}$ and $\frac{\partial f}{\partial p_2}$, we can compute them directly from function F. They are:

$$\frac{\partial f}{\partial p_1} = -\frac{1}{n-1} p_1^{-\frac{n}{n-1}} \tag{11}$$

$$\frac{\partial f}{\partial p_2} = e^{2\sigma_0(n)} \frac{1}{n-1} p_2^{-\frac{n}{n-1}} \tag{12}$$

By working backward, the differential tree can find p_1, p_2, $\frac{\partial p_1}{\partial r}$, $\frac{\partial p_2}{\partial r}$, $\frac{\partial p_1}{\partial v}$, $\frac{\partial p_2}{\partial v}$, respectively. So $\frac{\partial f}{\partial p_2}$ and $\frac{\partial f}{\partial p_2}$ are also available. By the Newton-Raphson method, we can finally evaluate r_{k+1} and v_{k+1}. With a good initial guess, the Newton-Raphson method can converge in just a few steps.

5 Fusion of Forward and Backward Induction

The algorithm presented in the previous section is a method based on backward induction. When we iterate to estimate the baseline rate and the v_i's at the

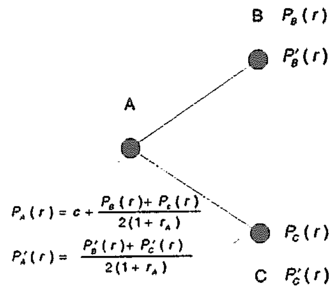

Figure 3: DIFFERENTIAL TREE. The simultaneous evaluation of a function and its derivative with the binomial tree structure. The cash flow at node A is denoted by c.

period, we sweep backwards several times through the tree. This is a time-consuming process. The running time is $O(n^3)$ since each sweeping takes $O(i^2)$ time. To remove the shortcoming, we will give an algorithm that runs in *quadratic* time based on forward induction. We continue to use the Newton-Raphson method to find the short rates. But now the idea is to sweep a line across time forward. Every time when calibrating the short rate tree, we proceed to compute the *discount factor* of each node at the next stage — that is, computing how much one dollar would contribute to the model price using such numbers from its two predecessors and the current short rate. This enhancement is consistent with the differential tree method; it simply speeds it up.

Let's be more precise. Suppose we are at the end of period i. So there are $i+1$ nodes. Let the baseline rate for period be r and $P_1, \ldots, P_i$ be the discount factors at the node of the previous period. By definition, $\sum_{j=1}^{i} P_j$ would be the model price for one dollar $i-1$ periods from today. We call a tree with these discount factors a *binomial discount factor tree*. We need to keep it in mind that there is no need to actually store the whole discount factor tree; just the current column suffices. The details can be found in Lyuu (1995). The running time is now quadratic.

6 Experimental Results

6.1 Efficiency and accuracy

The goal of the short rate tree is to evaluate interest rate-sensitive securities. To get accurate solutions for the values of the securities, we need a tree with finely spaced steps between today and the expiration. By our algorithm, it turns out to be easy to find the short rate tree from the given market term structure.

While implementing the algorithms, we are mainly concerned about the accuracy of the resulting short rate tree and time complexity. Since it is inconvenient to display the final resulting short rate tree, we just depict the final results in Figure 4 and the number of iterations accompanied by its standard deviation in Figure 5.

It is clear that the running time of the forward induction algorithm is far less than the time taken by the backward induction. Notice that both algorithms can calibrate huge trees due to the $O(n)$ representation of the tree allowed by the BDT model.

6.2 Discussions on the experiments

1. We ran the programs under Sun UltraSparc with 256MB of DRAM.
2. While solving the nonlinear equations using Newton-Raphson method, it is very important to get a suitable initial guess in order to reduce the running time. In our algorithm, we used the baseline rate and multiplicative ratio of the previous period as our initial guesses, and it worked well. By the above figures, we can solve for the baseline rate and multiplicative ratio in just a few iterations (four, in this case).
3. We take the upward-sloped yield curve scenario and declining volatility structure as our input. In particular, the current t-period pure discount yield is $0.06 + 0.005 \times \ln t$ and the t-period pure discount yield volatility is $1.4 \times (1 - e^{-0.1 \times t})/t$ (Gagnon and Johnson (1994) and Hull and White (1990)). "Unreasonable" term structures may have convergence problems.
4. We took the relative error at 10^{-13} in our program. This is a very stringent requirement.
5. We can build the short rate tree with one day steps (365 periods per year) and 30 year-horizon. Since

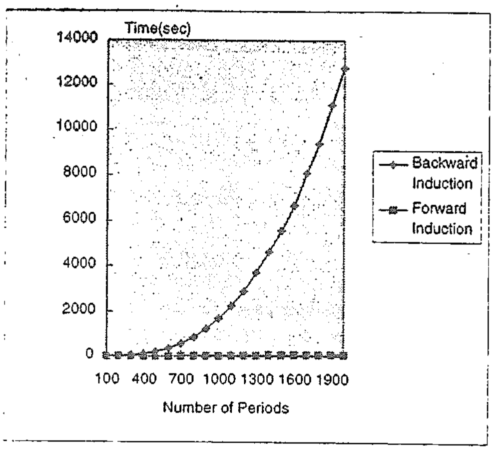

Figure 4: Time Efficiency of the Two Algorithms Compared.

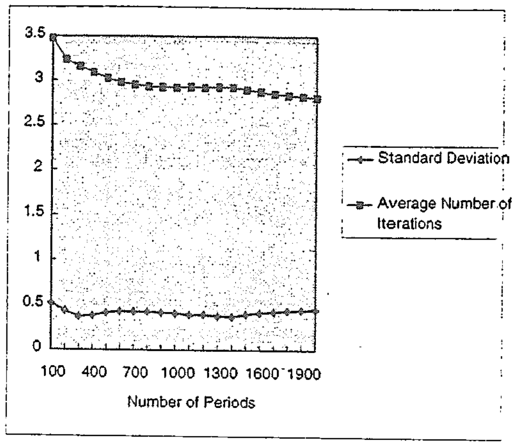

Figure 5: Average number of Iterations and Standard Deviation.

we just have to store the baseline rate and the multiplicative ratio, the memory required is just 2.2×10^4 instead of 6×10^7.

7 Conclusions

In this paper, we propose an approach to find the tree of short rates in the case of the BDT model. The approach uses backward induction with differential tree and forward induction procedures. We show how to implement both procedures and compare their time efficiency. The results show that forward induction is robust and much more numerically efficient than backward induction. The forward induction method can be viewed as an enhanced differential tree algorithm made possible by the special structure of the model.

Most of the people use the well-known backward induction procedure. But the paper shows it is more efficient to use the forward induction procedure due to the iterative nature of the root-finding process. Once the tree has been constructed, most interest rate-sensitive securities can be valued easily.

References

[1] Black, F., E. Derman, and W. Toy. "A One-Factor Model of Interest Rates and Its Application to Treasury Bond Options." *Financial Analysts Journal*, January–February 1990, pp. 33–39.

[2] Gagnon, L., and L.D. Johnson. "Dynamic Immunization under Stochastic Interest Rates." *The Journal of Portfolio Management*, 20, No. 3 (Spring 1994), 48–54.

[3] Hildebrand, F.B. *Introduction to Numerical Analysis.* 2nd ed. New York: Dover, 1974.

[4] Ho, T.S.Y., and S.B. Lee. "Term Structure Movements and Pricing Interest Rate Contingent Claims." *Journal of Finance*, 16, No. 5 (December 1986), 1011–1029.

[5] Hull, John C. *Options, Futures, and Other Derivative Securities.* 3rd ed. Englewood Cliffs, New Jersey: Prentice-Hall, 1997.

[6] Hull, J., and A. White. 'Pricing Interest Rate Derivative Securities." *The Review of Financial Studies*, 3 (1990), 573–592.

[7] Jamshidian, F. "Forward Induction and Construction of Yield Curve Diffusion Models." *The Journal of Fixed Income*, June 1991, pp. 62–74.

[8] Lyuu, Y.-D. *Introduction to Financial Computation: Principles, Mathematics, Algorithms.* Manuscripts, 1995–1997.