Binomial Term Structure Models

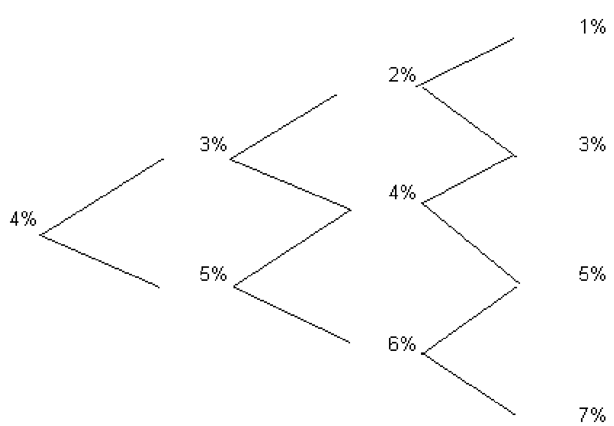
In this article, the authors develop several discrete versons of term structure models and study their major properties. We demonstrate how to program and calibrate such models as Black-Derman-Toy and Black-Karasinski. In addition we provide some simple methods for pricing options on interest rates.

by Simon Benninga and Zvi Wiener

The term structure models discussed in our previous article ("Term Structure of Interest Rates," *MiER* Vol. 7, No. 3) such as the Vasicek 1978] or the [Cox-Ingersoll-Ross 1985] model may not match the current term structure. The Black-Derman-Toy (BDT) and Black-Karasinksi models discussed in this article are important examples of models in which the current term structure can always be replicated.

1. BINOMIAL INTEREST RATE MODELS

Before introducing these models, we give a short introduction to binomial interest rate models. It is helpful to consider the following example. Suppose that one-period interest rates develop in a binomial model according to the following stochastic process:



In this example the interest rate process is as follows: The short-term interest rate today is 4% (for "short-term," read "one period"). In each succeeding period, the short-term interest rate either goes up or goes down by one percent.

Clearly, this process has some problems: For example, at some point interest rates will become negative, a highly undesirable property of a model which is supposed to describe nominal interest rates! For the moment we ignore this problem, and forge on with the example.

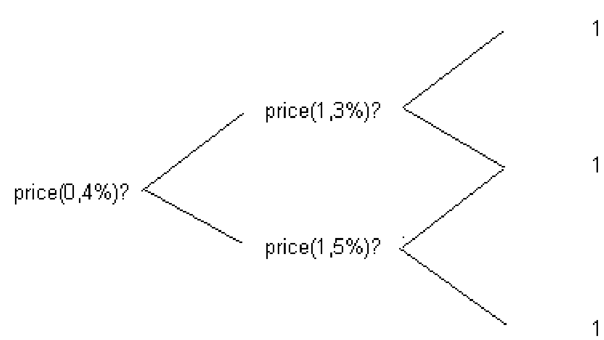
Risk-neutrality: using the model to calculate the term structure

To use our simple model for price calculations, we have to make two more assumptions:

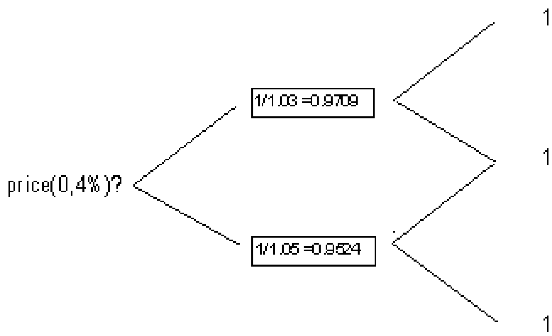
a. The probability of the interest rate going up or down from each node is 0.5.

b. The state probabilities can be used to do value calculations: The value of a state-dependent security is the *present value of its expected payoffs.*

Together, these assumptions are **the risk-neutrality assumption**. In this article we shall not justify this assumption on economic grounds. Here's what risk-neutrality means: Consider the problem of calculating the price at time 0 of a two-period, pure discount bond. Such a bond has payoffs only at date 2; with no loss in generality, we assume that these payoffs are $1:



The prices of the bond (as yet to be calculated) are indicated in the drawing above. "price(1,3%)" refers to the price of the bond at date 1 when the one-period interest rate is 3%. To calculate the prices, we first discount the bond payoffs to date 1:

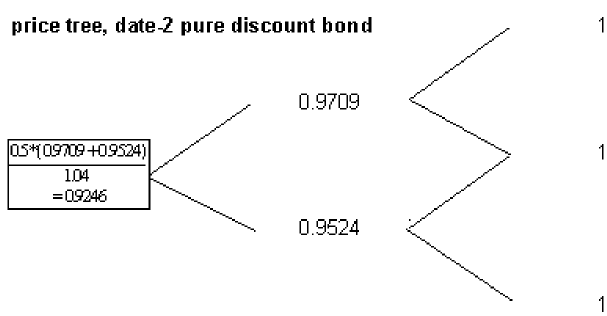


The problem now is to find the price today, *price(0,4%)*. Here's where risk neutrality comes in. Assuming risk neutrality, we can calculate

$$\texttt{In[1]:=}\ \mathbf{price[0,\,0.04]} = \frac{\mathbf{0.5 * price[1,\,0.03] + 0.5 * price[1,\,0.05]}}{\mathbf{1.04}}\ \texttt{/.}\ \{\mathbf{price[1,\,0.03]} \to \mathbf{1/1.03},\ \mathbf{price[1,\,0.05]} \to \mathbf{1/1.05}\}$$

$\texttt{Out[1]=}$ 0.924642

Thus the tree of prices for this date-two pure discount bond looks like:

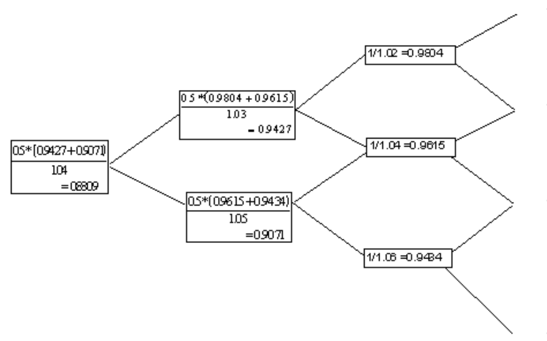


Note that *price(0,4%)* gives us the *two-date pure discount yield*:

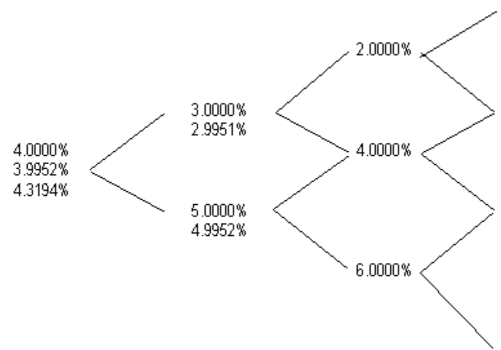
$$\texttt{In[2]:=}\ \mathbf{r_2} = \left(\frac{\mathbf{0.5} * \left(\frac{1}{1+r_{111}} + \frac{1}{1+r_{101}}\right)}{1 + r_1}\right)^{-0.5} - 1\ \texttt{/.}\ \{r_1 \to 0.04,\ r_{111} \to 0.05,\ r_{101} \to 0.03\}$$

$\texttt{Out[2]=}$ 0.0399519

(Here we have started to sneak in some notation: *rtjm* is the *m-period* interest rate at time *t* when the interest rate has made *j* "up" moves. Thus, in the above example, *r*001 = 4%, *r*111 = 5%, *r*101 = 3%, and we have now shown that *r*002 = 3.9952%.) Clearly, we could go on: Extending the tree by one date and considering a 3-date pure discount bond will allow us to calculate more interest rates. Here, for example, is the price tree for a date-3 pure discount bond:

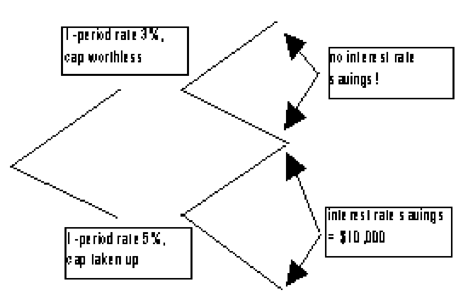


This price tree enables us to calculate the 3-period pure discount rate at date 0 (*r003*) and also the 2-period pure discount rates at date 1, *r*102 and *r*112:

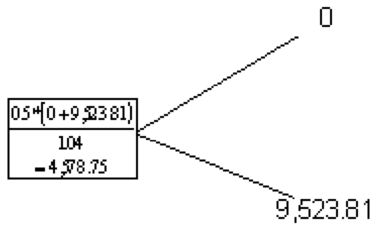


(The rates are listed under each other at each node, in increasing order: I.e., at date 0 the one-period rate is 4%, the two period rate is 3.9952%, the three-period rate is 4.3194%.)

Pricing options on the term structure: an example
We can also use this simple model to price options whose payoffs are functions of the term structure. Suppose, for example, that we are trying to price an **interest-rate cap**. This is a security which offers the borrower a loan at a guaranteed rate in the future. For this simple example, we consider a cap which offers the borrower a one-period loan of $1,000,000 at date 1 with a rate no higher than 4%. Here is a picture which explains it all:



When the cap is taken up (as you can see in the above picture, this only happens when the interest rate tomorrow is 5%), the savings are $\$10,000/1.05 = \$9,523.81$. Given risk-neutrality, we discount these savings to arrive at the value of the cap today:



Thus, if we were offered an option to get a one period loan tomorrow at an interest rate no higher than 4%, this option would be worth (today) \$4,578.75. Clearly we could use our model to price other, more complicated, derivative securities whose value depends on interest rates.

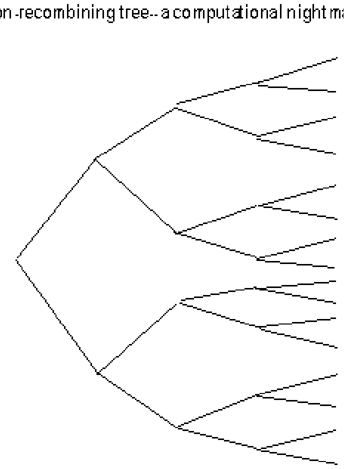
What is a "reasonable" interest rate tree?

The above example is an effective way of seeing why we would like a binomial model of interest rates. It illustrates some of the tools we will be using in the rest of the article, and it also illustrates some of the problems which we may encounter (it is clear, for example, that extending the tree for a couple of more dates will give us negative interest rates, an undesirable property).

To set the stage for the two models we will be discussing next, we state what we want from a good binomial interest rate model.

- Recombining in interest rates. In order to make computations easy, the interest rate which results from an "up-down" sequence should be equal to the interest rate resulting from a "down-up" sequence of moves. In principle, we could have an interest rate model which looks like the following picture, but this would give us severe computational problems.

Non-recombining tree--a computational nightmare!



- Non-negative interest rates. Assuming that the tree models nominal interest rates, we want these rates to be always positive. (Although we should add a caveat: Full-blown general equilibrium models of the term structure almost always model *real interest rates*, which are quite often negative.)
- Incorporates risk-neutrality. If we have risk-neutrality, then asset values are determined by discounting their expected future values. This means that prices and yields are easy to calculate.
- Replicates the current term structure of interest rates. At date 0 (today), the tree should give the currently observed yields for pure discount bonds.
- Replicates other "reasonable" properties of interest rates and interest rate-derivative securities. Some "reasonable" properties might include: a) The model replicates currently-observed cap prices. b) The model incorporates some mean-reversion of interest rates. In general we think we see that high rates tend to go down and vice versa.

2. THE BLACK-DERMAN-TOY MODEL

The BDT model is the simplest recombining interest rate model which replicates the current term structure. The insight from which the BDT model starts is the following: At any particular point in time we know the term structure of interest rates; for example, suppose that today:

one period pure discount rate, $r001$ = 10%,

two-period pure discount rate, $r002$ = 11%,

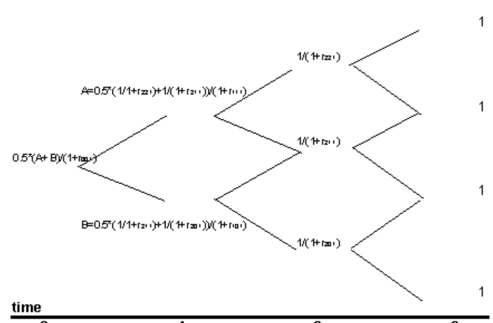
three-period pure discount rate, $r003$ = 12%,

four-period pure discount rate, $r004$ = 12.5%,

five-period pure discount rate, $r005$ = 13%.

Now suppose that we make the following assumptions:

- The interest rates "develop" in a binomial model. Each node of this model has a one-period interest rate attached to it. Our convention is to use $rtjm$ to represent the m-term interest rate at time t when there have been j "up" moves in the interest rate.
- The probabilities of the occurrence of the states in the model are always $\frac{1}{2}$.
- We have some other information about the interest rates (we will be more specific later).



To see why these assumptions are important, look at the above figure, in which we have indicated the one-period interest rates at each node (nodes are numbered by the time period and the number of “up” moves). The logic of this figure was explained in the previous section: The purpose here is to calculate the 3-period pure discount rate.

At time 3, the bond reaches maturity and has value 1, irrespective of the state of nature.

At time 2, the value of the bond is $1/(1 + r2j1)$, where j is the number of “up” moves.

At time 1, the value of the bond is the *expected discounted value* of its value one-period hence discounted at the one-period discount rate at time 2 (this rate is, of course, state dependent). Here we use risk neutrality and our assumption about the interest rates!

At time 0, the value of the bond is the expected discounted value of its time 1 value, discounted at $r001$, the one-period rate at time 0.

The development of the interest rate process in the BDT model

BDT assume that the interest rates at date t are defined by $r_{tj1} = r_{t01}e^{2j\theta_t}$. θ_t is the standard deviation of the natural logarithm of the one-period interest rates at date *t.* To see where this structure comes from, look at the one-period interest rates at time 1: $r101$ and $r111$. Assume that $r_{111} = r_{101}e^{2\psi}$. Then calculating the mean and the variance of the logarithm of the interest rates, we get:

mean log interest rate =

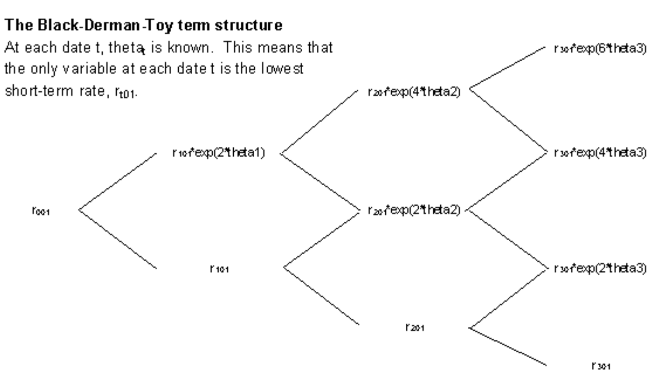
$$\frac{1}{2}[\log(r_{101}e^{\psi}) + \log(r_{101})] = log(r_{101}) + \frac{\psi}{2}$$

variance of log interest rate =

$$\frac{1}{2}\Big[\log(r_{101}e^{\psi}) - \Big(\log(r_{101}) + \frac{\psi}{2}\Big)\Big]^2 + \frac{1}{2}\Big[\log(r_{101}) - \Big(\log(r_{101}) + \frac{\psi}{2}\Big)\Big]^2 = \frac{\psi^2}{4}$$

so that *standard deviation of log interest rate* = $\frac{\psi}{2}$

BDT assume that the standard deviation of the log interest rates at any date t isθ_t. It thus follows that at date 1, $r_{111} = r_{101}e^{2\theta_1}$. Similarly, at any date t, we will have $r_{tj1} = r_{t01}e^{2j\theta_t}$.



The BDT assumption about the volatility of the short-term interest rates at any date t means that $rt01$ can be adjusted in order to fit the current term structure. Here is an example of how this works. The table below gives the term structure for years 1, 2, . . . , 5 as well as a list of s, one for each year.

maturity (years)	**yield to** maturity(%)	**volatility of one-**period rate = θ_t
1	10%	
2	11%	19%
3	12%	18%
4	12.5%	17%
5	13%	16%

3. CALIBRATING THE BDT MODEL

Given this information, we can calculate the term structure of the interest rates Suppose we know the risk-neutral probabilities, the current term structure and the volatility of the interest rates. The whole interest rate tree, in Excel, is shown below; after the picture of the tree we show you how to check that the interest rates are indeed correct:

	A	B	C	D	E	F	G	H	I
25	The Black-Derman-Toy interest rate tree								28.01%
26									
27									
28							22.84%		
29									
30					19.69%				20.34%
31									
32									
33			14.32%				16.26%		
34									
35					13.74%				
36	10%								14.77%
37									
38									
39			9.79%				11.57%		
40									
41									
42					9.59%				10.72%
43									
44									
45							8.24%		
46									
47									
48									7.79%

4. PROGRAMMING BLACK-DERMAN-TOY

Excel is a nice way to illustrate the BDT model, but *Mathematica* is much better for programming the model. The advantage of *Mathematica* is that it allows us to write the present value functions as recursive functions, which saves a lot of time and allows for much greater clarity.

First we write a function which takes the present value on the tree, assuming that the local interest rates are given by `r[t,j]`:

```
In[3]:= Clear[discount, pv, r]
        discount[r_] := 1/(1 + r); pv[n_] :=
           Module[{recurse}, recurse[t_, j_] :=
              recurse[t, j] =
               If[t == n, 1, discount[r[t, j]]*
                 (recurse[t + 1, j + 1] +
                  recurse[t + 1, j])];
           recurse[0, 0]/2^n]
```

pv[n] is the n-period discount factor. Here are some examples, to show you that the recursive function pv defines these discount factors correctly:

```
In[4]:= Print["pv[0] = ", pv[0]]
        Print["pv[1] = ", pv[1]]
        Print["pv[2] = ", pv[2]]
```

pv[0] = 1

$$\text{pv[1]} = \frac{1}{1+\text{r[0,0]}}$$

$$\text{pv[2]} = \frac{\frac{2}{1+\text{r[1,0]}} + \frac{2}{1+\text{r[1,1]}}}{4\,(1+\text{r[0,0]})}$$

If we now add the definition which is central to BDT:

```
In[5]:= r[t_, j_] := r[t] * Exp[2 * θ[t] * j];
```

Then we see that the present value factors are completely determined by the theta[1], theta[2], . . . and by r[0], r[1], . . . :

```
In[6]:= Print[pv[2] =, pv[2]]
```

$$\text{pv[2]} = \frac{\frac{2}{1+\text{r[1]}} + \frac{2}{1+e^{2\,\theta[1]}\,\text{r[1]}}}{4\,(1+\text{r[0]})}$$

You will recognize that:

r[0] is the one-period interest rate today.

r[1] is the lowest one-period interest rate at time $t = 1$. The other interest rate at time 1 is determined by theta[1], which is the θ of the interest rates at time $t = 1$, so that r[1,j] = r[1]*Exp[2*theta[1]*j], j = 0, 1, 2 ... . In general: r[n,j] = r[n]*Exp[2*theta[n]*j], j = 0, 1, ..., n

We can now solve for the BDT term structure. We have to know two things:

The term structure itself; we will denote this by R[1], R[2], ...

The theta[1], theta[2], ...

Here is the program:

```
In[7]:= Clear[discount, pv, r, theta]
        theta[1] = 0.19; theta[2] = 0.18;
        theta[3] = 0.17;
          theta[4] = 0.16; R[1] = 1/1.1;
          R[2] = 1/1.11^2; R[3] = 1/1.12^3;
          R[4] = 1/1.125^4; R[5] = 1/1.13^5;

          r[t_, j_] := r[t]*Exp[2*theta[t]*j];
          discount[r_] := 1/(1 + r);

          pv[n_] :=
            Module[{recurse}, recurse[t_, j_] :=
              recurse[t, j] =
               If[t == n, 1, discount[r[t, j]]*
                 (recurse[t + 1, j + 1] +
                  recurse[t + 1, j])];
              recurse[0, 0]/2^n]
```

To get the output:

```
In[8]:= Clear[rr]
        rr[0] = FindRoot[pv[1] == R[1],
                 {r[0], {0, 1}}]

        a = pv[2] == R[2] /. rr[0];
        rr[1] = FindRoot[a, {r[1], {0, 1}}]

        b = pv[3] == R[3] /. rr[0] /. rr[1];
        rr[2] = FindRoot[b, {r[2], {0, 1}}]

        c = pv[4] == R[4] /. rr[0] /.
              rr[1] /. rr[2];
        rr[3] = FindRoot[c, {r[3], {0, 1}}]

        d = pv[5] == R[5] /. rr[0] /. rr[1] /.
              rr[2] /. rr[3];
        rr[4] = FindRoot[d, {r[4], {0, 1}}]

        MatrixForm[Table[Table[rr[t][[1,2]]*
         Exp[2*theta[t]*j], {j, 0, t}], {t, 0, 4}]]
```

{r[0] → 0.1}

{r[1] → 0.0979156}

{r[2] → 0.0958616}

{r[3] → 0.0823614}

{r[4] → 0.0778718}

$$\begin{pmatrix} \{0.1\} \\ \{0.0979156, 0.14318\} \\ \{0.0958616, 0.137401, 0.196941\} \\ \{0.0823614, 0.115713, 0.162571, 0.228404\} \\ \{0.0778718, 0.107239, 0.147682, 0.203377, 0.280077\} \end{pmatrix}$$

The last lines of the output give the whole development of the term structure–all of the one-period interest rates.

5. SIMPLE ALTERNATIVES TO BLACK-DERMAN-TOY

Once you understand the logic of BDT, it is easy to see that there are many alternative binomial term structure models which might also work. In this section we present three such models.

5.a. Putting the lowest interest rate on top

In the BDT model the relation between interest rates at time t and time $t+1$ is weak, but not totally arbitrary. For example, suppose that we specify that the interest rates at time t, instead of developing as $r_{tj1} = r_{t01}e^{2\theta j}$, develop as $r_{tj1} = r_{t01}e^{-2\theta j}$. This means that the *bottom* interest rate in the tree at each time will be the highest instead of the lowest interest rate. The resulting interest rates will be the same as before. Here is the revised program and the output:

```
In[9]:= Clear[discount, pv, r, theta]
        theta[1] = 0.19; theta[2] = 0.18;
        theta[3] = 0.17; theta[4] = 0.16;
        R[1] = 1/1.1; R[2] = 1/1.11^2;
        R[3] = 1/1.12^3; R[4] = 1/1.125^4;
        R[5] = 1/1.13^5;
```

```
r[t_, j_] := r[t]*Exp[-2*theta[t]*j];
discount[r_] := 1/(1 + r);

pv[n_] :=
  Module[{recurse}, recurse[t_, j_] :=
    recurse[t, j] =
     If[t == n, 1, discount[r[t, j]]*
       (recurse[t + 1, j + 1] +
        recurse[t + 1, j])];
    recurse[0, 0]/2^n]
```

```
In[10]:= Clear[rr]
         rr[0] = FindRoot[pv[1] == R[1],
                 {r[0], {0, 1}}];
         a = pv[2] == R[2] /. rr[0];
         rr[1] = FindRoot[a, {r[1], {0, 1}}];
         b = pv[3] == R[3] /. rr[0] /. rr[1];
         rr[2] = FindRoot[b, {r[2], {0, 1}}];
         c = pv[4] == R[4] /. rr[0] /. rr[1] /.
                 rr[2];
         rr[3] = FindRoot[c, {r[3], {0, 1}}];
         d = pv[5] == R[5] /. rr[0] /. rr[1] /.
                 rr[2] /. rr[3];
         rr[4] = FindRoot[d, {r[4], {0, 1}}];
         MatrixForm[Table[Table[rr[t][[1,2]]*
           Exp[-2*theta[t]*j], {j, 0, t}],
             {t, 0, 4}]]
```

$$\begin{pmatrix} \{0.1\} \\ \{0.14318, 0.0979156\} \\ \{0.196941, 0.137401, 0.0958616\} \\ \{0.228404, 0.162571, 0.115713, 0.0823614\} \\ \{0.280077, 0.203377, 0.147682, 0.107239, 0.0778717\} \end{pmatrix}$$

5.b. A more radical change

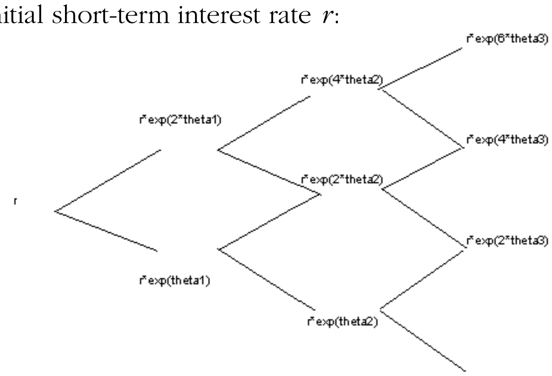
We can make the relation between dates more explicit by changing the model. For example, we could write:

$$\begin{cases} r_{t+1,j+1,1} = r_{tj1} e^{m_t+\theta}, \textit{ if interest rates move up} \\ r_{t+1,j,1} = r_{tj1} e^{m_t-\theta}, \textit{ if interest rates move down} \end{cases}$$

This guarantees that the interest rate tree is recombining. In this interest rate process: The interest rate *increments* θ are node and time independent, but the m_t, on the other hand, are time dependent but node independent. In this process the structure of the interest rate dynamics guarantees that interest rates will never be negative.

5.c. Another model

Another version of a recombining term structure model is to write the whole term structure as a function of the initial short-term interest rate r:



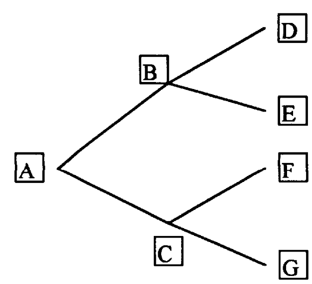
We leave the calculation of this model as an exercise.

6. THE BLACK-KARASINSKI MODEL

The BDT model may match the current term structure, but it does not have enough degrees of freedom to match other currently-observed market prices. For example, BDT may not match the prices of interest rate caps. It will certainly fail to capture the mean-reversion of interest rates.

The Black-Karasinski (BK) model aims to solve this problem by adding additional degrees of freedom to the interest rate process. The cost–as we shall see–is that the model's time structure is somewhat different from that of a standard binomial interest rate model.

In the BK model we assume that the stochastic process which defines the interest rates is given by $d(\log r) = \phi(t)[\log \mathrm{p}(t) - \log r]dt + \sigma(t)dz$. BK refer to $\mathrm{p}(t)$ as the *target rate*, $\phi(t)$ as the *mean reversion*, and $\sigma(t)$ as the *local volatility* of $\log r$. Consider the following tree, and let's consider what it would take to make this tree recombining in the BK model:



$$\log A = \log r$$
$$\log B = \phi(1)\left[\log\mu(1) - \log r\right]\Delta t_1 + \sigma(1)\sqrt{\Delta t_1} + \log r$$
$$\log C = \phi(1)\left[\log\mu(1) - \log r\right]\Delta t_1 - \sigma(1)\sqrt{\Delta t_1} + \log r$$
$$\log D = \phi(2)\left[\log\mu(2) - \log B\right]\Delta t_2 + \sigma(2)\sqrt{\Delta t_2} + \log B$$
$$\log E = \phi(2)\left[\log\mu(2) - \log B\right]\Delta t_2 - \sigma(2)\sqrt{\Delta t_2} + \log B$$
$$\log F = \phi(2)\left[\log\mu(2) - \log C\right]\Delta t_2 + \sigma(2)\sqrt{\Delta t_2} + \log C$$
$$\log G = \phi(2)\left[\log\mu(2) - \log C\right]\Delta t_2 - \sigma(2)\sqrt{\Delta t_2} + \log C$$

In order for the tree to be recombining, we must have E = F. Substituting B into E and C into F gives:

$$\begin{aligned} E &= \phi(2)\left[\log\mu(2) - \log B\right]\Delta t_2 - \sigma(2)\sqrt{\Delta t_2} + \log B \\ &= \phi(2)\left[\log\mu(2) - \phi(1)\left[\log\mu(1) - \log r\right]\Delta t_1 - \sigma(1)\sqrt{\Delta t_1} - \log r\right]\Delta t_2 - \sigma(2)\sqrt{\Delta t_2} \\ &\qquad + \phi(1)\left[\log\mu(1) - \log r\right]\Delta t_1 + \sigma(1)\sqrt{\Delta t_1} + \log r \\ F &= \phi(2)\left[\log\mu(2) - \log C\right]\Delta t_2 + \sigma(2)\sqrt{\Delta t_2} + \log C \\ &= \phi(2)\left[\log\mu(2) - \phi(1)\left[\log\mu(1) - \log r\right]\Delta t_1 + \sigma(1)\sqrt{\Delta t_1} - \log r\right]\Delta t_2 + \sigma(2)\sqrt{\Delta t_2} \\ &\qquad + \phi(1)\left[\log\mu(1) - \log r\right]\Delta t_1 - \sigma(1)\sqrt{\Delta t_1} + \log r \end{aligned}$$

Setting these two expressions equal and cleaning up gives:

$$\phi(2)\left[-\sigma(1)\sqrt{\Delta t_1}\right]\Delta t_2 - \sigma(2)\sqrt{\Delta t_2} + \sigma(1)\sqrt{\Delta t_1} = \phi(2)\left[1+\sigma(1)\sqrt{\Delta t_1}\right]\Delta t_2 + \sigma(2)\sqrt{\Delta t_2} - \sigma(1)\sqrt{\Delta t_1}$$

$$\sigma(1)\sqrt{\Delta t_1} = \phi(2)\left[\sigma(1)\sqrt{\Delta t_1}\right]\Delta t_2 + \sigma(2)\sqrt{\Delta t_2}$$

$$\phi(2)\left[\sigma(1)\sqrt{\Delta t_1}\right]\Delta t_2 = \sigma(1)\sqrt{\Delta t_1} - \sigma(2)\sqrt{\Delta t_2} \Rightarrow \phi(2) = \frac{1}{\Delta t_2}\left[1 - \frac{\sigma(2)\sqrt{\Delta t_2}}{\sigma(1)\sqrt{\Delta t_1}}\right]$$

This last equation is BK's equation (3). What this equation shows is that the *length* of the interval *t2* and the mean reversion $\phi(2)$ are functions of each other. Black-Karasinski claim that the solution to this equation is:

$$Black-Karasinski\ \ solution = \Delta t_2 = \frac{4\Delta t_1\left(\frac{\sigma(2)}{\sigma(1)}\right)^2}{\left(1+\sqrt{1+4\phi(2)\left(\frac{\sigma(2)}{\sigma(1)}\right)^2\Delta t_1}\right)}$$

We do not prove this directly, but rather use version 3 of *Mathematica* to show that this is the correct solution (we need version 3 because it contains the command `//FullSimplify`, which is more powerful than version 2's `//Simplify`.) We first use `Solve` to give us a solution to the equation $\phi(2) = \frac{1}{t_2}\left[1 - \frac{\sigma(2)\sqrt{t_2}}{\sigma(1)\sqrt{t_1}}\right]$ in terms of t_2.

```
In[11]:= sol = Solve[phi2 - 1/delta2
              (1 - 1/s * Sqrt[delta2/delta1]) == 0,
            delta2]
```

Out[11]=

$$\left\{\left\{\text{delta2} \to \frac{1 + 2\ \text{delta1 phi2 s}^2 - \sqrt{1 + 4\ \text{delta1 phi2 s}^2}}{2\ \text{delta1 phi2}^2\ \text{s}^2}\right\}, \left\{\text{delta2} \to \frac{1 + 2\ \text{delta1 phi2 s}^2 + \sqrt{1 + 4\ \text{delta1 phi2 s}^2}}{2\ \text{delta1 phi2}^2\ \text{s}^2}\right\}\right\}$$

The two solutions of this equation are given below:

```
In[12]:= s1=sol[[1,1,2]]
         s2=sol[[2,1,2]]
```

Out[12]=

$$\frac{1 + 2\ \text{delta1 phi2 s}^2 - \sqrt{1 + 4\ \text{delta1 phi2 s}^2}}{2\ \text{delta1 phi2}^2\ \text{s}^2}$$

$$\frac{1 + 2\ \text{delta1 phi2 s}^2 + \sqrt{1 + 4\ \text{delta1 phi2 s}^2}}{2\ \text{delta1 phi2}^2\ \text{s}^2}$$

The Black-Karasinski solution is written as:

```
In[13]:= BKSoln=
          delta1*4*s^2/
           (1+Sqrt[1+4*phi2*s^2*delta1])^2
```

Out[13]=

$$\frac{4\ \text{delta1 s}^2}{\left(1 + \sqrt{1 + 4\ \text{delta1 phi2 s}^2}\right)^2}$$

We now use `FullSimplify` to show that `s1` is the same as the BK solution, whereas `s2` is not (this is very inelegant, but it works):

```
In[14]:= BKSoln - s1//FullSimplify
Out[14]= 0
```

```
In[15]:= BKSoln - s2//FullSimplify
```

Out[15]=

$$-\frac{\sqrt{1 + 4\ \text{delta1 phi2 s}^2}}{\text{delta1 phi2}^2\ \text{s}^2}$$

What does the BK solution mean? A first illustration

The Black-Karasinksi solution means that as *n* gets larger, the number of divisions of a given time period gets correspondingly larger. We give 2 illustrations, both based on numbers from BK. We assume that `phi1=phi2= ... = 0.1` and that `s = 1` (this means that all σ(i) are constant). The BK solution can be written as a recursive function:

```
In[16]:= Clear[δ]
         δ[0] = 1;
         δ[n_] := δ[n] = FractionBox()
```

$$\frac{4 * \delta[n-1] * s^2}{\left(1 + \sqrt{1 + 4 * \delta[n-1] * \text{phi2} * s^2}\right)^2}\ \text{/. \{phi2} \to 0.1,\ s \to 1\}$$

A table of the first 10 Δt shows that they get progressively smaller:

```
In[17]:= Table[δ[j], {j, 0, 10}]
Out[17]= {1, 0.839202, 0.722343, 0.633694, 0.564205,
          0.508305, 0.462385, 0.424006, 0.391459,
          0.363516, 0.339269}
```

What does the BK solution mean? A second illustration

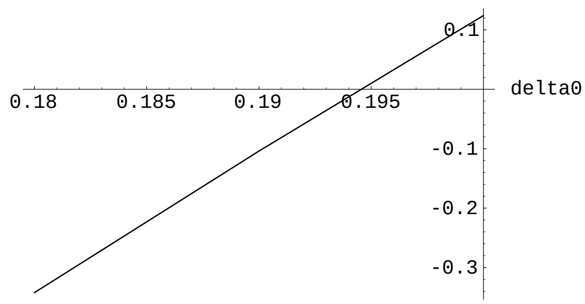
BK have another way of illustrating this time dependence of Δt: Suppose we want to divide a 10-year period into 160 subperiods. How big should our initial t_0 be so that 10 years will be covered exactly by 160 subperiods?

To solve this problem, we first redefine our function `delta`, to make it dependent on the initial *t*0 (which we call, of course, `delta0`):

```
In[18]:= Clear[delta, time]
         delta[n_, delta0_] :=
           delta[n, delta0] =
            If[n == 0, delta0,
              4*delta[n - 1, delta0]*
               s^2/(1 + Sqrt[1 + 4*phi*s^2*
                delta[n - 1, delta0]])^2 /.
              {phi -> 0.1, s -> 1}]
         time[n_, delta0_] :=
           time[n, delta0] =
             Sum[delta[j, delta0], {j, 0, n - 1}]
```

The function `time` gives the total time elapsed (i.e., the sum of all the `delta`), given the initial `delta0`. Here is a plot of `time` for a range of `delta0`:

```
In[19]:= data =
          Table[{delta0, time[160, delta0] - 10},
             {delta0, 0.18, 0.2, 0.01}];
         ListPlot[data,
           PlotJoined -> True,
           AxesLabel -> {delta0, ""}];
```



The solution is to put the initial `delta` somewhere between 0.194 and 0.195. To solve exactly, we use a variation of the bisection routine we first illustrated in the article on finding the implied Black-Scholes option pricing volatility:

```
In[20]:= Module[{high, low}, high = 0.2;
            low = 0.19;
            While[Abs[time[160, (high + low)/2] -
              10] > 0.00001,
             Print["(high + low)/2 = ",
               (high + low)/2];
             Print["time[160, (high + low)/2] = ",
                    time[160, (high + low)/2]];
              If[time[160, (high + low)/2] > 10,
                 high = (high + low)/2,
               low = (high + low)/2]];
            Print[delta0 = N[(high + low)/2]]]
```

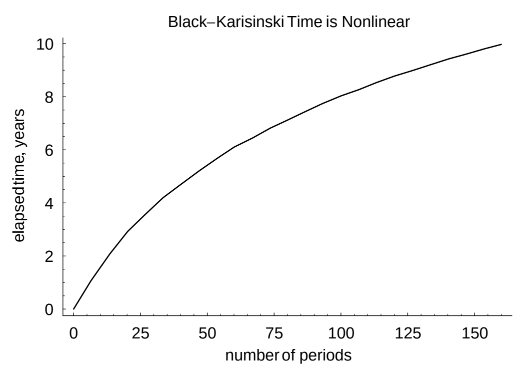
```
(high + low)/2 = 0.195
time[160, (high + low)/2] = 10.0112
(high + low)/2 = 0.1925
time[160, (high + low)/2] = 9.95391
(high + low)/2 = 0.19375
time[160, (high + low)/2] = 9.98264
(high + low)/2 = 0.194375
time[160, (high + low)/2] = 9.99694
(high + low)/2 = 0.1946875
time[160, (high + low)/2] = 10.0041
(high + low)/2 = 0.194531
time[160, (high + low)/2] = 10.0005
(high + low)/2 = 0.194453
time[160, (high + low)/2] = 9.99873
(high + low)/2 = 0.194492
time[160, (high + low)/2] = 9.99962
(high + low)/2 = 0.194512
time[160, (high + low)/2] = 10.0001
(high + low)/2 = 0.194502
time[160, (high + low)/2] = 9.99985
(high + low)/2 = 0.194507
time[160, (high + low)/2] = 9.99996
(high + low)/2 = 0.194509
time[160, (high + low)/2] = 10.
(high + low)/2 = 0.194508
time[160, (high + low)/2] = 9.99999
0.194509
```

This particular routine prints out intermediate results; the last result shows that `delta0 = 0.194509` gives ten years. Notice that the last line of the above routine also *names* `delta0`, (we do this by writing `Print[delta0= N[(high+low)/2]]`). ... (Not all the output printed)

Time in the BK model

Black-Karasinksi time is non-linear! Suppose, for example, we continue the BK example, dividing a 10 year period into 160 intervals. As we have seen above, this means that the length of the first interval is 0.194509; as we also showed above, the intervals get progressively shorter. This means that more and more periods are needed for a specific time interval, as shown in the graph below:

```
In[21]:= Plot[time[n, delta0], {n, 0, 160},
            FrameLabel ->
             {"number of periods",
              "elapsed time, years"},
            Frame -> {True, True, False, False},
            DefaultFont -> {"Helvetica", 8},
            PlotLabel ->
             "Black-Karisinski Time is Nonlinear"];
```



BK (in Table 1 of their paper) have yet another way of illustrating this: How many years have passed after 1/5 of the intervals (i.e., 32 periods) have elapsed? After 2/5 of the intervals? Clearly the answer is given by our *Mathematica* function `time[32,delta0]`, `time[64, delta0]`, etc.:

```
In[22]:= time[32,delta0]
         time[64,delta0]
         time[96,delta0]
         time[128,delta0]
```

```
4.10683
6.33608
7.87391
9.04894
```

7. CALCULATING A BLACK-KARASINSKI TERM STRUCTURE

Suppose we try to calculate a BK term structure for the following parameter values:

initial interest rate = 8%

mean reversion $\phi(t)$ = 4 (i.e., not time dependent)

standard deviation $\sigma(t)$ = 0.05 (also not time dependent)

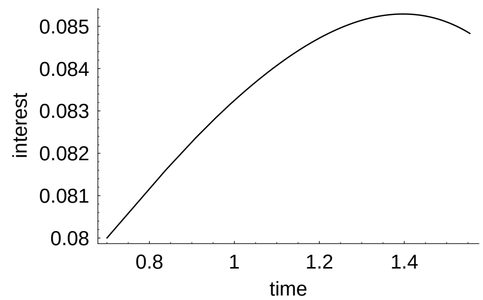
target interest rate p(t) = 10% (time independent)

The following *Mathematica* program calculates the term structure:

```
Clear[f, g, delta, time, yield, discount]
$RecursionLimit = 1000; target = 0.1;
phi = 4; sigma = 0.05; initial = 0.08;
  delta0 = 0.7; delta[n_] :=
   delta[n] = If[n == 0, delta0,
     4*delta[n - 1]/
      (1 + Sqrt[1 + 4*phi*delta[n - 1]])^2]
time[n_] :=
  time[n] = Sum[delta[j], {j, 0, n - 1}]
f[0, 0] := Log[initial]; f[n_, j_] :=
   f[n, j] = If[n == 0, f[0, 0],
     If[j >= 1, phi*(Log[target] -
       f[n - 1, j - 1])*delta[n] +
       sigma*Sqrt[delta[n]] + f[n - 1, j - 1],
      phi*(Log[target] - f[n - 1, 0])*delta[n] -
     sigma*Sqrt[delta[n]] + f[n - 1, 0]]]
g[n_, j_] := Exp[f[n, j]]
discount[n_, j_] :=
  discount[n, j] =
   If[n == 0, 1, 0.5*discount[n - 1, j - 1]*
     Exp[-g[n - 1, j - 1]*delta[n - 1]]]
yield[n_] := yield[n] =
   Log[Sum[Binomial[n, j]*discount[n, j],
     {j, 0, n}]]/-time[n]
```

We can now use the program to plot a sample term structure:

```
a = Table[{time[h], yield[h]}, {h, 1, 350}];
  ListPlot[a, PlotJoined -> True,
  AxesOrigin -> {0, 0.08}, PlotRange -> All,
   Frame -> {True, True, False, False},
   DefaultFont -> {"Helvetica", 10},
   FrameLabel -> {time, interest},
   PlotLabel ->
    StyleForm[
     "A Black-Karasinski Term Structure \n",
     "Section"]];
```



REFERENCES

Black, Fisher and Piotr Karasinski (1991). Bond and Option Pricing when Short Rates are Lognormal. *Financial Analysts Journal* (July-August), pp. 52-59.

Black, Fisher, Emmanual Derman, and William Toy (1990). A One-Factor Model of Interest Rates and its Application to Treasury Bond Options. *Financial Analysts Journal* (January-February).

Hull, John and A. White (1990). Pricing Interest Rate Derivative Securities. *Review of Financial Studies* 3, pp. 573-592.

ABOUT THE AUTHORS

The authors acknowledge grants from Wolfram Research, the Krueger and Eshkol Centers at the Hebrew University, and the Israeli Academy of Science. Wiener's research has benefited from a grant from the Israel Foundations Trustees, the Alon Fellowship and the Eshkol grant.

Simon Benninga is professor of finance at Tel-Aviv University (Israel) and the Wharton School of the University of Pennsylvania. He is the author of *Financial Modeling* (MIT Press, 1997) and of *Corporate Finance: A Valuation Approach* (with Oded Sarig, McGraw-Hill, 1997); he is also the editor of the *European Finance Review*.

Simon Benninga
Faculty of Management
Tel-Aviv University, Tel-Aviv, Israel
benninga@post.tau.ac.il
http://finance.wharton.upenn.edu/~benninga

Zvi Wiener is assistant professor of finance at the business school of the Hebrew University of Jerusalem. His finance research concentrates on the pricing of derivative securities, value-at-risk, computational finance and stochastic dominance. He wrote this article while visiting at the Olin School of Business at Washington University in St. Louis.

Zvi Wiener
Finance Department, Business School
Hebrew University, Jerusalem, Israel
mswiener@mscc.huji.ac.il
http://pluto.mscc.huji.ac.il/~mswiener/zvi.html