

# How does the Interest Rate Volatility depend on the level of the Interest Rate?

Thomas J. Mather

Department of Mathematics, Princeton University, Princeton, NJ

May 4th, 1998

## Abstract

This paper investigates the dependence of the interest rate volatility on the level of the interest rate. It considers a class of models which are Markov with respect to two state variables. In this family, the interest rate volatility of the form  $\sigma r^\gamma$ . This paper attempts to find the  $\gamma$  which best explains a series of cap prices.

## 1 Introduction

This paper considers a family of models for pricing interest rate derivatives. These models belong to the class of Heath, Jarrow, and Morton (HJM) (1992) models in that the initial forward rate curve and volatility structures are inputs to the model. This class of models was developed by Ritchken and Sankarasubramanian (hereafter RS) (1995).

## 2 Interest rate dynamics under the Heath-Jarrow-Morton Framework

The HJM framework models the evolution of the instantaneous forward rate,  $f(t, T)$ , which is defined by

$$f(t, T) = -\frac{\partial}{\partial T} \log P(t, T)$$

where  $P(t, T)$  denotes the price at time  $t$  of a bond which has a payoff of one dollar at  $T$ , the time of maturity.

Forward rates in a one-factor HJM framework are assumed to follow a stochastic differential equation of the form

$$df(t, T) = \mu^f(\omega, t, T)dt + \sigma^f(\omega, t, T)dz(t)$$

where  $z(t)$  is standard Brownian motion, and  $\omega$  represents the history of the Brownian motion. If there is no arbitrage, then under an equivalent risk-neutral martingale measure, HJM show that

$$\mu^f(\omega, t, T) = \sigma^f(\omega, t, T) \int_t^T \sigma^f(\omega, t, u)du$$

It follows that the evolution of the forward rates is given by the SDE

$$df(t, T) = \sigma^f(\omega, t, T) \int_t^T \sigma^f(\omega, t, u) du dt + \sigma^f(\omega, t, T) dz(t) \quad (1)$$

Under the HJM framework, the dynamics of the term structure are completely determined by the forward rate volatility  $\sigma^f(\omega, t, T)$  and the initial forward rate curve  $f(0, t)$ .

This paper considers volatility structures of the form  $\sigma r^\gamma e^{-\int_t^T \kappa(\tau) d\tau}$ . It can be shown<sup>1</sup>, that under this condition, that the spot rate dynamics are given by

$$dr = \mu^r(r, \phi, t)dt + \sigma r^\gamma dz(t) \quad (2)$$

$$d\phi = \mu^\phi(r, \phi, t)dt \quad (3)$$

with

$$\mu^r(r, \phi, t) = \kappa(t)(f(0, t) - r) + \phi(t) + \frac{df(0, t)}{dt} \quad (4)$$

$$\mu^\phi(r, \phi, t) = \sigma^2 r^{2\gamma} - 2\kappa(t)\phi(t) \quad (5)$$

$$(6)$$

and that forward rate curve is given by

$$f(t, T, r, \phi) = \phi\xi(t, T) \int_t^T \xi(t, \tau) d\tau - \xi(t, T)(f(0, t) - r) + f(0, T)$$

### 3 A Lattice Method for the Two State Variable Process

The model considered in this paper is Markov with respect to two state variables. In order to value it on a recombining lattice, it is necessary to model the evolution of both  $r$  and  $\phi$ . Most of the other lattices used to model interest rates only model a single state variable  $r$ . To model the second state variable,  $\phi$ , a lattice approximation was considered by Li, Ritchkin and Sankarasubramanian (1995) for  $\kappa(t)=\kappa$  and  $\gamma = 0.5$  as well as  $\gamma = 1$ . In this section, this lattice is generalized for all  $\kappa(t)$  and  $\gamma$ .

To build a lattice, it is helpful to have a process that has constant volatility. Therefore the following transformation is considered for  $\gamma \neq 1$ :

$$y(t) = \int \frac{1}{\sigma m^\gamma} dm = \frac{1}{\sigma} \frac{1}{1 - \gamma} r^{1-\gamma}$$

The corresponding inverse transform is

$$r(t) = (\sigma(1 - \gamma)y)^{\frac{1}{1-\gamma}}$$

The evolution of  $y$  is given by Itô's lemma

$$dy = \mu^y(y, \phi, t)dt + dz(t) \quad (7)$$

$$d\phi = \mu^\phi(y, \phi, t)dt \quad (8)$$

---

<sup>1</sup>see Mather (1998) pp. 3-6. with  $g(\phi, t, T) = \phi\xi(t, T) \int_t^T \xi(t, \tau) d\tau - \xi(t, T)f(0, t) + f(0, T)$

where, for  $\gamma \neq 1$  and  $r > 0$ ,

$$\begin{aligned}\mu^y(y, \phi, t) &= \frac{\partial y(t)}{\partial t} + \frac{\partial y(t)}{\partial r} \mu^r(r, \phi, t) + \frac{1}{2} \frac{\partial^2 y(t)}{\partial r^2} \sigma^2 r^{2\gamma} \\ &= \frac{1}{\sigma r^\gamma} \left( \kappa(t) (f(0, t) - r) + \phi(t) + \frac{df(0, t)}{dt} \right) - \frac{1}{2} \frac{\gamma}{\sigma r^{\gamma+1}} \sigma^2 r^{2\gamma} \\ &= \frac{1}{\sigma} (\sigma(1 - \gamma)y)^{-\frac{\gamma}{1-\gamma}} \left( \kappa(t) f(0, t) + \phi(t) + \frac{df(0, t)}{dt} \right) - \kappa(t)(1 - \gamma)y - \frac{1}{2} \frac{\gamma}{1 - \gamma} \frac{1}{y}\end{aligned}$$

and

$$\begin{aligned}\mu^\phi(y, \phi, t) &= \sigma^2 r^{2\gamma} - 2\kappa(t)\phi(t) \\ &= \sigma^2 (\sigma(1 - \gamma)y)^{\frac{1}{1-\gamma}} - 2\kappa(t)\phi(t)\end{aligned}$$

If  $\gamma = 1$ ,

$$y(t) = \int \frac{1}{\sigma m} dm = \frac{\ln r}{\sigma}$$

The corresponding inverse transform is

$$r(t) = e^{\sigma y}$$

and the structures for equations (7-8) are given by

$$\begin{aligned}\mu^y(y, \phi, t) &= \frac{\partial y(t)}{\partial t} + \frac{\partial y(t)}{\partial r} \mu^r(r, \phi, t) + \frac{1}{2} \frac{\partial^2 y(t)}{\partial r^2} \sigma^2 r^2 \\ &= \frac{1}{\sigma r} \left( \kappa(t) (f(0, t) - r) + \phi(t) + \frac{df(0, t)}{dt} \right) - \frac{1}{2} \sigma \\ &= \frac{e^{-\sigma y}}{\sigma} \left( \kappa(t) f(0, t) + \phi(t) + \frac{df(0, t)}{dt} \right) - \frac{\kappa(t)}{\sigma} - \frac{1}{2} \sigma\end{aligned}$$

and

$$\begin{aligned}\mu^\phi(y, \phi, t) &= \sigma^2 r^2 - 2\kappa(t)\phi(t) \\ &= \sigma^2 e^{2\sigma y} - 2\kappa(t)\phi(t)\end{aligned}$$

A binomial lattice approximation for  $y$  and  $\phi$  can now be obtained. This procedure begins by dividing the time interval into subintervals of length  $\Delta t$ . Let  $y(t, j)$  represent the value of  $y$  at time interval  $t$  after  $j$  up moves in the lattice. In the next time increment,  $y$  can either move up to  $y(t + 1, j + 1)$  or down to  $y(t + 1, j)$ . Since  $y$  has a constant volatility of 1,

$$\begin{aligned}y(t + 1, j + 1) &= y(t, j) + \sqrt{\Delta t} \\ y(t + 1, j) &= y(t, j) - \sqrt{\Delta t}\end{aligned}$$

Let  $\bar{\phi}(t, j)$  be the maximum value of  $\phi$  at time  $t$  with  $y = y(t, j)$ , maximized over all possible paths leading to  $y(t, j)$ . Similarly, let  $\underline{\phi}(t, j)$  be the minimal value of  $\phi$ , minimized over all possible paths leading to  $y(t, j)$ . These values are given by

$$\begin{aligned}\bar{\phi}(t + 1, j) &= \bar{\phi}(t, j) + \mu^\phi(y, \phi, t) \Delta t \\ \underline{\phi}(t + 1, j) &= \underline{\phi}(t, j - 1) + \mu^\phi(y, \phi, t) \Delta t\end{aligned}$$

Strike Rate	1 year	2 years	3 years	4 years	5 years	10 years
3.0	0.29011	0.62113	0.94180	1.25136	1.54915	2.83966
3.5	0.24261	0.52927	0.80882	1.08101	1.34493	2.50420
4.0	0.19524	0.43775	0.67707	0.91352	1.14545	2.18192
4.5	0.14798	0.34690	0.54770	0.75070	0.95296	1.87608
5.0	0.10084	0.25754	0.42251	0.59490	0.77013	1.58979
5.5	0.05423	0.17159	0.30431	0.44929	0.60025	1.32637
6.0	0.01705	0.09998	0.20468	0.32558	0.45502	1.09722
6.5	0.00605	0.06132	0.14246	0.24255	0.35314	0.92050
7.0	0.00184	0.03684	0.09877	0.18114	0.27533	0.77628
7.5	0.00043	0.02178	0.06843	0.13589	0.21594	0.65815
8.0	0.00007	0.01279	0.04757	0.10261	0.17053	0.56093
8.5	0.00001	0.00748	0.03319	0.07796	0.13554	0.48034
9.0	0.00000	0.00436	0.02324	0.05959	0.10835	0.41307
9.5	0.00000	0.00252	0.01634	0.04579	0.08709	0.35659
10.0	0.00000	0.00145	0.01153	0.03538	0.07035	0.30894
10.5	0.00000	0.00083	0.00817	0.02748	0.05709	0.26854
11.0	0.00000	0.00048	0.00581	0.02144	0.04654	0.23415

Table 1: Interest Rate Cap Prices

For each time  $t$  and  $y$ , partition  $\phi$  into  $m$  equidistant points  $\{\phi(t, j, 0), \phi(t, j, 1), \dots, \phi(t, j, m-2), \phi(t, j, m-1)\}$  with

$$\bar{\phi}(t, j) = \phi(t, j, 0) < \phi(t, j, 1) < \dots < \phi(t, j, m-2) < \phi(t, j, m-1) = \underline{\phi}(t, j)$$

The probabilities for  $y$  moving up or down are determined by the drift term for  $y$ ,  $\mu^y(y, \phi, t)$ . Let  $p(t, j, k)$  be the probability of  $y$  jumping up, where  $y = y(t, j)$  and  $\phi = \phi(t, j, k)$ . Then

$$p(t, j, k) = \frac{1}{2}\mu^\phi(y, \phi, t)\sqrt{\Delta t} + \frac{1}{2}$$

Let  $\xi(t, j)$  be the payoff function of the interest rate claim. Then the value of the claim at time step  $t$ ,  $g(t, j, k)$ , is given by

$$g(t, j, k) = e^{-r\Delta t}(p(t, j, k)\bar{g}(t, j, k) + (1 - p(t, j, k))\underline{g}(t, j, k)) + \xi(t, j)$$

where  $\bar{g}(t, j, k)$ , the value of the claim if  $y$  jumps up, and  $\underline{g}(t, j, k)$ , the value of  $y$  goes down are given by

$$\begin{aligned}\bar{g}(t, j, k) &= (\bar{k} - \lfloor \bar{k} \rfloor)g(t+1, j+1, \lfloor \bar{k} \rfloor + 1) + (1 - \bar{k} + \lfloor \bar{k} \rfloor)g(t+1, j+1, \lfloor \bar{k} \rfloor) \\ \underline{g}(t, j, k) &= (\underline{k} - \lfloor \underline{k} \rfloor)g(t+1, j, \lfloor \underline{k} \rfloor + 1) + (1 - \underline{k} + \lfloor \underline{k} \rfloor)g(t+1, j, \lfloor \underline{k} \rfloor)\end{aligned}$$

where

$$\bar{k} = \frac{\phi(t, j, k) + \mu^\phi(y, \phi, t) - \underline{\phi}(t+1, j+1)}{\bar{\phi}(t+1, j+1) - \underline{\phi}(t+1, j+1)}$$

maturity	yield
spot rate	0.05500
1 year	0.05925
2 years	0.06265
3 years	0.06435
4 years	0.06545
5 years	0.06645
6 years	0.06725
7 years	0.06795
8 years	0.06865
9 years	0.06925
10 years	0.06985

Table 2: Treasury Bill Yield Curve

$$\underline{k} = \frac{\phi(t, j, k) + \mu^\phi(y, \phi, t) - \underline{\phi}(t + 1, j)}{\overline{\phi}(t + 1, j) - \underline{\phi}(t + 1, j)}$$

The present value of the claim,  $g(0, 0, 0)$ , can be found by backward recursion.

## 4 Pricing Interest Rate Caps

One of the more popular interest rate options is the interest rate cap. They provide insurance against the interest rate rising above a certain strike rate,  $K$ . The owner of a cap receives at the end of each quarter

$$\frac{1}{4} \max(R - K, 0)$$

where  $R$  is the three-month yield at the beginning of the quarter.

In this paper, cap prices on Treasury Bill yields provided by the Department of the Treasury were used<sup>2</sup>. This data, as given in tables 1 and 2 reflects the prices on June 30, 1997.

## 5 Results

Using the model developed in this paper, the market yield and cap prices were fitted, using the several different values of  $\gamma$ . There are several parameters that need to be specified when running this model. These include  $\kappa(t)$ ,  $\sigma$ ,  $\gamma$  and  $f(0, t)$ .  $\kappa(t)$  was set to 0.02 to account for mean reversion.  $f(0, t)$  was derived from the market yield as given in table 2. Different values of  $\gamma$ ,  $\{\gamma = 0.05, 0.06, \dots, 0.14, 0.15\}$ , were tested. For each value of  $\gamma$ , the  $\sigma$  that best fit<sup>3</sup> the cap prices

<sup>2</sup>see <http://www.ots.treas.gov/salpt/table017.html>

<sup>3</sup>in other words, the value of  $\sigma$  that minimized the distance between the market prices and those given by the model

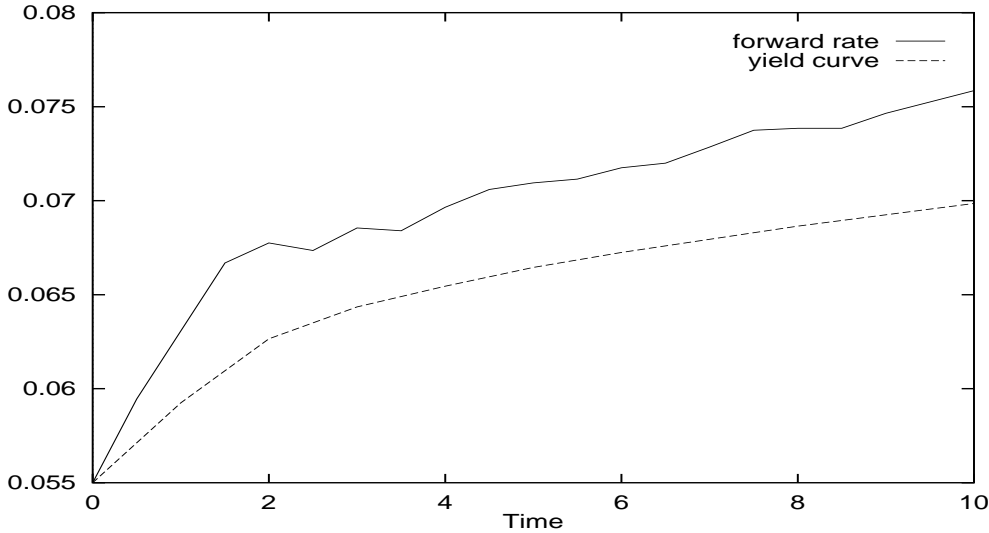


Figure 1: Yield curve vs. Forward rate curve

in table 1 was found using the golden section search algorithm<sup>4</sup>. The distance<sup>5</sup> between the cap prices given by the model and the actual market prices is given by table 3 for each  $\gamma$  and maturity.

The smaller the distance, or “goodness of fit”, the better the model of the interest rate dynamics is. For maturities of 1 and 2 years, values of  $\gamma = 1.3$  best explains the market prices. For maturities of between 3 and 5 years, a value of  $\gamma = 1.2$  is implied. Finally,  $\gamma = 1.0$  best explains the market prices over ten years.

These results are consistent with those obtained by Chan, Karolyi, Longstaff and Sanders (1992), who used the Generalized Method of Moments to estimate  $\gamma$ . Their unconstrained estimate was  $\gamma = 1.499$  for short term yields. However, their study was based on *historical* estimate of  $\gamma$  given by one-month yields for Treasury bills from 1964 to 1989, while this paper studies the *implied*  $\gamma$  given by the market’s *expectation* of future interest rate movements.

## 6 Conclusion

In this paper, the sensitivity of the interest rate volatility to the level of the interest rate was estimated. The results of this study show that, in order to fit interest rate cap prices, the interest rate volatility must be high dependent on the level of the interest rate. The implied value of  $\gamma$ , which measures how much the interest rate volatility is dependent on the level of the interest rate, was found to be between 1.0 and 1.3, depending on the maturity considered.

This study is of particular importance, since one of the most widely used models, the Hull-White Model, assumes that the interest rate volatility is independent of the interest rate level, that is,  $\gamma = 0$ . Therefore, it will perform poorly relative to other models such as Black-Derman-Toy ( $\gamma = 1$ ), and the model considered in this paper. The advantage of the model considered in this

<sup>4</sup>see Press, Teukolsky, Vetterling, Flannery, *Numerical Recipes in C* (1992) pp.397-402

<sup>5</sup>as measured by the sum over strike prices of the proportional difference squared between the actual cap price and the price given by the model

$\gamma$	1 year	2 years	3 years	4 years	5 years	10 years
0.5	0.078265	0.285638	0.239071	0.123707	0.094049	0.051368
0.6	0.066792	0.215870	0.178421	0.091419	0.067854	0.033820
0.7	0.057081	0.155254	0.127117	0.063827	0.046800	0.021129
0.8	0.049216	0.106833	0.084921	0.041969	0.029648	0.011329
0.9	0.043682	0.067863	0.051040	0.025051	0.016726	0.004043
1.0	0.040087	0.038332	0.026248	0.012779	0.007551	<b>0.001251</b>
1.1	0.038362	0.017912	0.010281	0.005248	0.002938	0.003672
1.2	0.038283	0.007712	<b>0.002801</b>	<b>0.002329</b>	<b>0.001998</b>	0.014600
1.3	<b>0.037648</b>	<b>0.005832</b>	0.003682	0.004041	0.005711	0.036253
1.4	0.038009	0.011627	0.012886	0.018497	0.013780	0.070747
1.5	0.039355	0.025161	0.030245	0.088885	0.026173	0.126912

Table 3: Distance, or “goodness of fit” as a function of  $\gamma$ . The smallest distance is highlighted in bold for each maturity.

paper is that the  $\gamma$  can be specified to be any real number, and unlike Black-Derman-Toy model, the interest rate process is stationary.

## Appendix

This is the code used for the model considered in this paper.

```

/* hjm.c an interest rate pricing model. Copyright 1998, Thomas J. Mather
   Version 0.9alpha Last revision date May 04, 1998
   How deos the Interest Rate Volatility depend on the level of the Interest
   Rate? Working Paper - http://www.princeton.edu/~tmather/
   Thomas J. Mather '99 tjmather@alumni.princeton.edu */
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<assert.h>
// Delta t = 1/(4*INTERVAL)
#define INTERVAL 4
// SPAN = Time Horizon = T
#define SPAN 3
// PARTITION = number of phi at each node = m
#define PARTITION 3
typedef struct node2 *node;
struct node2 { double y; double maxphi; double minphi; };
double hjm(double *forward, double *dforward, double *kappa, double sigma,
           int n, double length, int partition, double gamma);
////////////////////////////////////

```

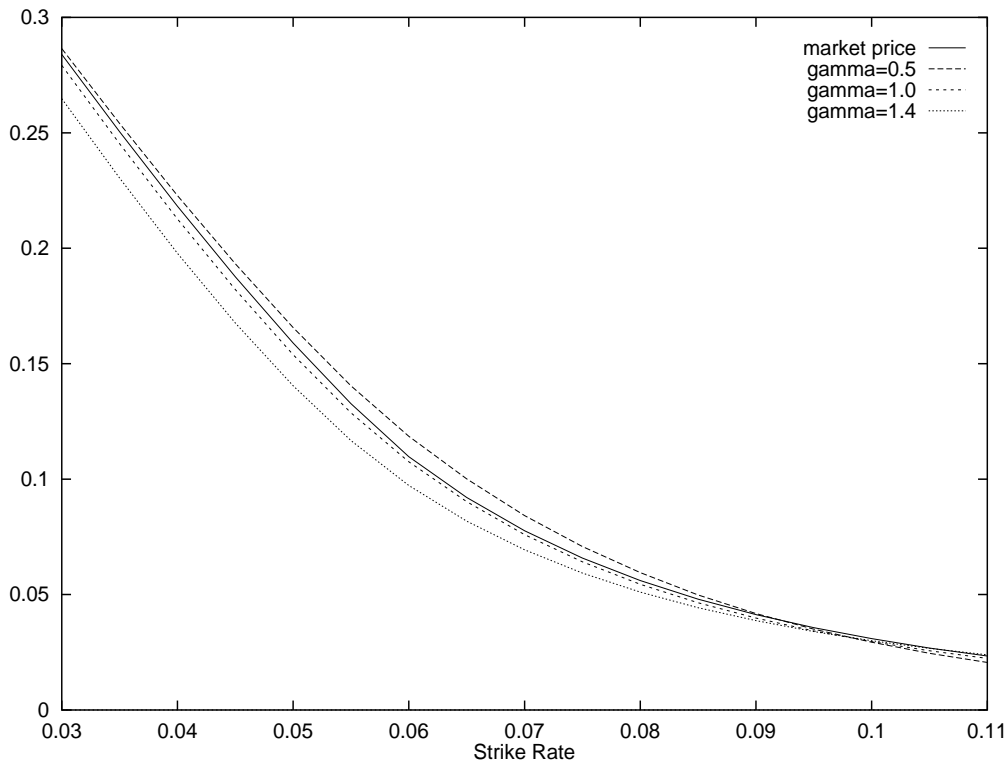


Figure 2: The value of interest caps for different  $\gamma$

```
// main: Initializes Term Structure. Find the optimal value of sigma
// using the Golden Section Search.
void main()
{
    int i,tint1,tint2;

    // This sets the number of periods to be considered
    int period=SPAN*4*INTERVAL+1;
    int partition=PARTITION;
    double gamma;

    // This is a second parameter which measures the degree of
    // mean reversion it can be time dependent
    double kappa2[11];
    double kappa[SPAN*4*INTERVAL+1];
    double sigma1, sigma2, sigma3, midsigma;

    // added 0.8
    // this gives the 0 to 10 year market yields
    double marketyield[11]={0.055,0.05925,0.06265,0.06435,0.06545,0.06645,
    0.06725,0.06795,0.06865,0.06925,0.06985};
```

```

double marketdyield[10];

//      this gives estimated market forward rate
double marketforward[10];
double marketfor[21];

//      this gives the estimated derivative of the forward rate
double marketdfor[20];

//      this is the intial forward rate curve.
//      it is found from the value of interest rate swaps
double forward[SPAN*4*INTERVAL+1];

//      the derivative of the forward rate
double dforward[SPAN*4*INTERVAL+1];

//      length gives the length of the partition in years
double length=(double)1/(4*INTERVAL);
double value,midvalue,value2,t,a;

for(i=0;i<10;i++)
marketdyield[i]=marketyield[i+1]-marketyield[i];
for(i=0;i<11;i++)
kappa2[i]=0.02;

//      kappa2[0]=-0.6;
//      kappa2[1]=-0.8;
//      kappa2[2]=0.1;
//      kappa2[3]=-0.752;
//      kappa2[4]=0.97;

marketfor[0]=marketyield[0];

//      This constructs the forward curve, using fitting a forward rate
//      "spline" to the market yield
for(i=1;i<10;i++)
marketfor[2*i]=marketyield[i]+i*(marketdyield[i-1]+marketdyield[i])/2;
marketfor[20]=marketyield[10]+10*marketdyield[9];
for(i=0;i<10;i++)
{
marketforward[i]=(marketyield[i+1]*(i+1)-marketyield[i]*i);
}
for(i=0;i<10;i++)
{
marketfor[2*i+1]=2*marketforward[i]-

```

```

(marketfor[2*i+2]+marketfor[2*i])/2;
}

//      this is the first time derivative of the forward rate
for(i=0;i<20;i++)
{
    marketdfor[i]=(marketfor[i+1]-marketfor[i])*2;
}

//      the values of the forward rate and its derivative are determined
//      using linear interpolation
for(i=0;i<period;i++)
{
    t=i*length;
    tint1=(int)(t*2);
    a=t-(float)tint1/2;
    forward[i]=(1-2*a)*marketfor[tint1]+2*a*marketfor[tint1+1];
    dforward[i]=marketdfor[tint1];
    kappa[i]=kappa2[tint1/2];
}

//      Various values of gamma are tested
for(gamma=1.5001;gamma<1.51;gamma+=0.1)
{

//      The sigma giving the minimum goodness-to-fit error is bracketed
//      by sigma=0.1 and sigma=0.3
    sigma1=.1;sigma2=.2;sigma3=.3;
    value2=hjm(forward, dforward, kappa, sigma2*pow(marketyield[0],1-gamma),
period, length,partition,gamma);
    for(i=0;i<20;i++)
    {

//      Golden Section Search
        midsigma=(sigma1+sigma2)/2;
        midvalue=hjm(forward, dforward, kappa,
midsigma*pow(marketyield[0],1-gamma), period,
length,partition,gamma);
        if(midvalue<value2)
        {
            sigma3=sigma2;
            sigma2=midsigma;
            value2=midvalue;
        }
        else

```

```

    {
        sigma1=midsigma;
        sigma2=(sigma2+sigma3)/2;
        value2=hjm(forward, dforward, kappa,
sigma2*pow(marketyield[0],1-gamma),
period, length,partition,gamma);}
    }
    printf("%f %f %f\n",gamma,midsigma, midvalue);
}
}
////////////////////////////////////////////////////
//          yfr returns y as a function of r
double yfr(double r, double sigma, double gamma)
{
    if(r<0){r=-r;}
    return (1/(1-gamma))*pow(r,1-gamma)/sigma;
}
////////////////////////////////////////////////////
//          rfy returns r as a function of y
double rfy(double y, double sigma, double gamma)
{
    if((1-gamma)*y<0){y=-y;}
    if(pow(sigma*(1-gamma)*y,1/(1-gamma))>40)
{
    return 40;
}
    else
return pow(sigma*(1-gamma)*y,1/(1-gamma));
}
////////////////////////////////////////////////////
//          muphi returns the drift coefficient for phi
double muphi(double r, double phi, double kappa, double sigma,double gamma)
{
    if(r<0){r=-r;}
    return sigma*sigma*pow(r,2*gamma)-2*kappa*phi;
}
////////////////////////////////////////////////////
//          muy returns the drift coefficient for y
double muy(double y, double phi, int i, double kappa, double sigma,
double *forward, double *dforward, double gamma)
{
    if((1-gamma)*y<0){y=-y;}
    return pow(sigma*(1-gamma)*y,-gamma/(1-gamma))*
(kappa*forward[i]+phi+dforward[i])/sigma-
kappa*(1-gamma)*y-gamma/(2*(1-gamma)*y);
}

```

```

}
//////////////////////////////////////*
// If gamma=1, then use these functions instead
double yfr(double r, double sigma,double gamma)
{
    return log(r)/sigma;
}
double rfy(double y, double sigma,double gamma)
{
    return exp(y*sigma);
}
double muphi(double r, double phi, double kappa, double sigma, double gamma)
{
    return sigma*sigma*r*r-2*kappa*phi;
}
double muy(double y, double phi, int i, double kappa, double sigma,
    double *forward, double *dforward, double gamma)
{
    return ((kappa*(forward[i]-exp(sigma*y))+phi+dforward[i])*exp(-sigma*y)-
    sigma*sigma/2)/sigma;
}
*/
//////////////////////////////////////
// This is the payoff of a call option as a function of r3, the
// 3 month yield
double payoff(double r,double r3,double strike)
{
    if(r3>strike)
    {
        return exp(-r3*0.25)*0.25*(r3-strike);
    }
    else
    return 0;
}
//////////////////////////////////////
// hjm2 takes the structure for y, minphi, maxphi and computes the
// value of an interest rate claim by using backward recursion
double hjm2(double *forward, double *dforward, double *kappa, double sigma,
    int n, double length, int partition, double **y, double **minphi,
    double **maxphi, double gamma)
{
    int i,j,k,phiindex,paymentdate;
    double ***price;
    double p,phi,nextphi,phiupweight,tmp,priceup,pricedown,strike,l;

```

```

////////////////////////////////////
//These are the market price for the interest rate caps
/* Term = 1 year */
// double marketcap[17]={0.029011,0.024261,0.019524,0.014798,0.010084,0.005423,
0.001705,0.000605,0.000184,0.000043,0.000007,0.000001,
0,0,0,0,0};
/* Term = 2 years */
// double marketcap[17]={0.062113,0.052927,0.043775,0.034690,0.025754,0.017129,
0.009998,0.006132,0.003684,0.002178,0.001279,0.000748,
0.000436,0.000252,0.000145,0.000083,0.000048};
/* Term = 3 years */
// double marketcap[17]={0.094180,0.080882,0.067707,0.054770,0.042251,0.030431,
0.020468,0.014246,0.009877,0.006843,0.004757,0.003319,
0.002324,0.001634,0.001153,0.000817,0.000581};
/* Term = 4 years */
// double marketcap[17]={0.125136,0.108101,0.091352,0.075070,0.059490,0.044929,
0.032558,0.024255,0.018114,0.013589,0.010261,0.007796,
0.005959,0.004579,0.003538,0.002748,0.002144};
/* Term = 5 years */
// double marketcap[17]={0.154915,0.134493,0.114545,0.095296,0.077013,0.060025,
0.045502,0.035314,0.027533,0.021594,0.017053,0.013554,
0.010835,0.008709,0.007035,0.005709,0.004654};
/* Term = 10 years */
// double marketcap[17]={0.283966,0.250420,0.218192,0.187608,0.158979,0.132637,
0.109722,0.092050,0.077628,0.065815,0.056093,0.048034,
0.041307,0.035659,0.030894,0.026854,0.023415};
////////////////////////////////////
double error=0,marketprice;

///// Allocate memory
price=malloc(n*sizeof(double **));
for(i=0;i<=n;i++){
price[i]=malloc((i+1)*sizeof(double *));
for(j=0;j<=i;j++){
price[i][j]=malloc(partition*sizeof(double));
}
}

//// For each strike from 0.03 to 0.11 ... ////
for(strike=0.03;strike<=0.11001;strike+=.005)
{
for(j=0;j<=n;j++)
for(k=0;k<partition;k++)
price[n][j][k]=0;
for(i=n-1;i>=0;i--){

```

```

paymentdate=0;
l=floor(i*4*length);

////////// If at beginning of quarter, payoff occurs //////////
if((i*4*length)-l<0.0001&&i>-1){
    paymentdate=1;
}
for(j=0;j<=i;j++)
    for(k=0;k<partition;k++)
    {

////////// Calculate the probability of an "up" jump //////////
    phi=minphi[i][j]+k*(maxphi[i][j]-minphi[i][j])/
        (partition-1);
        nextphi=phi+muphi(rfy(y[i][j],sigma,gamma),phi,
            kappa[i],sigma,gamma)*length;
        p=muy(y[i][j],phi,i,kappa[i],sigma,forward,dforward,
            gamma)*sqrt(length)/2+.5;

////////// Calculate the price after an "up" jump //////////
        if(i==j)
        {
            priceup=price[i+1][j+1][0];
        }
        else
        {
            tmp=(partition-1)*(nextphi-
                minphi[i+1][j+1])/
            (maxphi[i+1][j+1]-minphi[i+1][j+1]);
            phiindex=(int)tmp;
            phiupweight=tmp-phiindex;
            priceup=(1-phiupweight)*
            price[i+1][j+1][phiindex]+
            (phiupweight)*price[i+1][j+1]
            [phiindex+1];
        }

////////// Calculate the price after a "down" move //////////
        if(j==0)
        {
            pricedown=price[i+1][j][0];
        }
        else
        {
            tmp=(partition-1)*(nextphi-

```

```

        minphi[i+1][j])/
(maxphi[i+1][j]-minphi[i+1][j]);
    phiindex=(int)tmp;
    phiupweight=tmp-phiindex;
    pricedown=(1-phiupweight)*
price[i+1][j][phiindex]+
(phiupweight)*price[i+1][j]
[phiindex+1];
}

////////// Calculate the current price from the
////////// "Pricedown" and "Priceup"
    price[i][j][k]=exp(-rfy(y[i][j],sigma,gamma)*length)*
    (p*pricedown+(1-p)*priceup)+paymentdate*
    payoff(rfy(y[i][j],sigma,gamma),
    rfy(y[i][j],sigma,gamma)+dforward[i]/4,strike);
}
}

////////// Calculate the "goodness of fit" error
    marketprice=marketcap[(int)(strike*200+0.001)-6];
//    printf("gamma: %f strike: %f price: %f market: %f error: %f\n",gamma,
//    strike,price[0][0][0],marketprice,
//    pow((marketprice-price[0][0][0])/price[0][0][0],2));
//        if(marketprice>0.000005)
error+=pow((marketprice-price[0][0][0])/price[0][0][0],2);
}

////////// Deallocate memory
    for(i=0;i<=n;i++)
{
    free(y[i]);
    free(minphi[i]);
    free(maxphi[i]);
}
    free(y);
    free(maxphi);
    free(minphi);
    for(i=0;i<=n;i++)
{
    for(j=0;j<=i;j++)
{
    free(price[i][j]);
}
}

```

```

    free(price[i]);
}
    free(price);
    return error;
}
////////////////////////////////////
// hjm calculates the values of y, maxphi, and minphi
double hjm(double *forward, double *dforward, double *kappa, double sigma,
    int n, double length, int partition, double gamma)
{
    int i,j,k;
    double **y;
    double **minphi;
    double **maxphi;

    //////////// Allocate memory for y, minphi, and maxphi
    y=malloc(n*sizeof(double *));
    minphi=malloc(n*sizeof(double *));
    maxphi=malloc(n*sizeof(double *));
    for(i=0;i<n;i++)
    {
        y[i]=malloc((i+1)*sizeof(double));
        minphi[i]=malloc((i+1)*sizeof(double));
        maxphi[i]=malloc((i+1)*sizeof(double));
    }

    y[0][0]=yfr(forward[0],sigma,gamma);
    maxphi[0][0]=0;
    minphi[0][0]=0;
    for(j=1;j<n;j++){

// deals with the boundary of the tree, where there is only
// one value of phi
maxphi[j][0]=minphi[j][0]=minphi[j-1][0]+muphi(rfy(y[j-1][0],sigma,gamma),
    minphi[j-1][0],kappa[j],
    sigma,gamma)*length;
    maxphi[j][j]=minphi[j][j]=minphi[j-1][j-1]+muphi(rfy(y[j-1][j-1],sigma,gamma),
    minphi[j-1][j-1],kappa[j],
    sigma,gamma)*length;

    //////////// Calculate y
    for(i=0;i<=j;i++)
    {
y[j][i]=y[0][0]+(j-2*i)*sqrt(length);
    }
}

```

```

////////// Calculate phi
for(i=1;i<j;i++)
{
maxphi[j][i]=maxphi[j-1][i]+muphi(rfy(y[j-1][i],sigma,gamma),
maxphi[j-1][i],kappa[j],sigma,
gamma)*length;
minphi[j][i]=minphi[j-1][i-1]+muphi(rfy(y[j-1][i-1],sigma,gamma),
minphi[j-1][i-1],kappa[j],sigma,
gamma)*length;
}
}
return hjm2(forward,dforward,kappa,sigma,n-1,length,partition,y,minphi,maxphi,gamma);
}

```

## References

- Black, F., E. Derman, and W. Toy. "A One Factor Model of Interest Rates and Its Application to Treasury Bond Options." *Financial Analysts Journal*, 46 (1990), 33-39.
- Chan, K., G. Karolyi, F. Longstaff, A. Sanders. "An Empirical Comparison of Alternative Models of the Short-Term Interest Rate" *The Journal of Finance* 47 (1992), 1209-1228
- Heath, D., R. Jarrow, and A. Morton. "Bond Pricing and the Term Structure of Interest Rates: A New Methodology for Contingent Claim Valuation." *Econometrica*, 60 (1992), 77-105.
- Hull, J., and A. White. "Pricing Interest Rate Derivative Securities" *Reviews of Financial Studies*, 3,4 (1990), 573-92.
- Li, A., P. Ritchken, and L. Sankarasubramanian, "Lattice Models for Pricing American Interest Rate Claims." *Journal of Finance*, 50 (1995), 719-37.
- Mather, T. "Two-State Markovian Representations of Term Structure Dynamics." Junior Paper, Princeton University, Princeton, NJ (1998)
- Ritchken, P., and L. Sankarasubramanian, "Volatility Structures of Forward Rates and the Dynamics of the Term Structure." *Mathematical Finance*, 5 (1995), 55-72.