Indian Institute of Management Calcutta

Working Paper Series

WPS No. 777
March 2016

Volatility curve generation using the Heston Model

Krishna Praturi
PGDM Student, IIM Calcutta, D. H. Road, Joka P.O., Kolkata 700 104 India

Binay Bhushan Chakrabarti
Professor, Indian Institute of Management Calcutta
D. H. Road, Joka, P.O. Kolkata 700 104
http://facultylive.iimcal.ac.in/workingpapers

Volatility curve generation using the Heston Model

Working Paper

by

a) Corresponding author: B.B.Chakrabarti

Prof. (Retd.) of Finance, IIM Calcutta, bbc@iimcal.ac.in

b) Krishna Praturi

PGDM student, IIM Calcutta, email: krishnap2016@email.iimcal.ac.in

I. Abstract

The objective of the paper is to generate a volatility curve for the USDINR currency pair based on the Heston model. For any tenor, I will be using the ATM implied volatility, the 25 Delta Risk Reversal implied volatility, and the 25 Delta Butterfly implied volatility as inputs. I intend to use MATLAB as the programming tool. The aim is to compare the generated volatility curve with the observed market volatility curve and consequently examine the efficacy of the Heston model.The paper will be confined to the USDINR currency market and the stochastic Heston model for the purposes of volatility curve generation.

This working paper is structured as follows. We first introduce the FX variant of the Black Scholes model and typical quoting conventions in the FX options market. We then illustrate the Heston model and the methodology of the calibration process. We then summarize the results obtained from the calibration and analyze the success of the Heston model in replicating the actual implied volatility curve observed in the USDINR market. MATLAB was used for simulating the pricing and calibration routines demonstrated in this paper; the relevant code is presented in the appendix.

II. Introduction

Options play a crucial role in the financial markets. The earliest model to provide a framework to price options was the Black Scholes model. It makes several assumptions, the most crucial one being that the underlying asset follows geometric Brownian motion with constant volatility. However, it has been empirically established over the past few decades that implied volatilities for different strikes are almost never constant. This disparity is called the volatility skew or smile. Generally, it is observed that at-the money options have lower implied volatilities than in-the-money or out-of-the-money options.

To account for this discrepancy, option pricing models began to model volatility as a stochastic process. All such pricing models are broadly referred to as “Stochastic Volatility Models”. The most widely used SV model is the Heston model. There are two key reasons behind the popularity of the Heston Model. Firstly, it is better at predicting the shape of the implied volatility curve than the Black Scholes Model. Secondly, the Heston model is easier to implement than more complicated SV models. However the Heston model is limited in that it fails to predict far in-the-money or far out-of-the-money option implied volatilities. This can be mainly attributed to that fact that we use at-the-money or close to at-the-money volatilities to calibrate the Heston model parameters. I intend on using the Heston model to generate a volatility curve for the USDINR currency pair.

The FX Market

It is general convention in most markets to quote option prices at different strikes, but the FX market is unique in the sense that quotes provided in this market are implied volatilities at different deltas. The standard quotes are the at-the-money (ATM) volatility, 25Delta Risk Reversal and the 25Delta Butterfly;

for a better calibration of the volatility surface, the 10Delta Risk Reversal and the 10Delta Butterfly. For the purposes of completion, I now define some regularly used terms in this paper.

*Volatility Curve:*It is a plot of the implied volatility as a function of either the strike price or the delta. In the FX market and therefore for the purposes of this working paper, we plot the implied volatility as a function of the delta.

Risk Reversal: It is the difference between the implied volatility of the call price and the implied volatility of the put price at a specified moneyness level. For example, the 25 Delta Risk Reversal would be the difference between the implied volatility of the 25 Delta call and the implied volatility of the 25 Delta put.

$$25D\ RR = \sigma_{25C} - \sigma_{25P}$$

Butterfly: It is the difference between the average volatility of a call and put option at a specified moneyness level and the implied volatility of an ATM option.

$$25D\ BF = \frac{\sigma_{25C} + \sigma_{25P}}{2} - \sigma_{ATM}$$

Given the ATM volatility and the risk reversal, butterfly for a specific delta, the implied volatility for a call/put option at that delta can be calculated:

$$\sigma_{25C} = \sigma_{25D\ BF} + \frac{\sigma_{25D\ RR}}{2} + \sigma_{ATM}$$

$$\sigma_{25P} = \sigma_{25D\ BF} - \frac{\sigma_{25D\ RR}}{2} + \sigma_{ATM}$$

The typically quoted volatilities in the interbank market are the ATM volatility, 25D and 10D Risk Reversal and the Butterfly. Using these, it is possible to construct the implied volatility surface using a SV model.

The Garman Kohlhagen Model:

This is the variant of the Black-Scholes model used in pricing FX options given by the following equations.

$$C = Se^{-r_f t}N(d_1) - Ke^{-r_d t}N(d_2)$$

$$P = Ke^{-r_d t}N(-d_2) - Se^{-r_f t}N(-d_1)$$

$$d_1 = \frac{\log\left(\frac{S}{K}\right) + \left(r_d - r_f + \frac{\sigma^2}{2}\right)t}{\sigma\sqrt{t}}$$

$$d_2 = d_1 - \sigma\sqrt{t}$$

IV. The Heston Model

Heston proposed that the stock price and the volatility follow the below given processes:

$$dS_t = \mu S_t dt + \sqrt{\vartheta_t} S_t dW_t^1$$

$$d\vartheta_t = \kappa(\theta - \vartheta_t) + \sigma\sqrt{\vartheta_t} dW_t^2$$

$$\rho dt = dW_t^1 dW_t^2$$

Recognize that σ here does not represent the volatility itself but rather the volatility of the variance and ϑ_t represents the variance of the underlying. The positive value for σ is responsible for the smile in the volatility curve and a nonzero correlation ρ generates a skew in the volatility curve.

Parameters in this model:

θ Long term mean of variance

ϑ_t Variance of the underlying at time t

σ Volatility of the variance

This parameter affects the kurtosis of the underlying asset price distribution. When σ is 0, there is no kurtosis, as σ increases, so does the kurtosis of the distribution. At high values of σ, there is a greater chance of extreme market movements.

ρ Correlation between the two Weiner processes

If ρ>0, the volatility increases as the stock price increases. If ρ<0, the volatility increases as the stock price decreases. When ρ =0, there is no skewness to the distribution.

κ Rate of mean reversion for the variance

This parameter represents how fast the variance reverts to its long term mean. In a way, it also signifies the degree of "volatility clustering", viz., the likelihood that large price variations will be followed by large price variations.

The closed form solution to the Heston Model for European FX options is as follows: (ω =1 for call, ω =-1 for put, u_1= 0.5, u_2=-0.5)

$$h(t) = \omega[S_t e^{-r_f t} P_+(\omega) - K e^{-r_d t} P_-(\omega)]$$

$$P_+(\omega) = \frac{1-\omega}{2} + \omega P_1(logS_t, \vartheta_t, \tau, logK)$$

$$P_{-}(\varphi) = \frac{1-\omega}{2} + \omega P_2(logS_t, \vartheta_t, \tau, logK)$$

$$P_j(x, \vartheta_t, \tau, y) = \frac{1}{2} + \frac{1}{\pi}\int_0^{\infty} \frac{Re(e^{-i\varphi y} f_j(x, \vartheta_t, \tau, \varphi)}{i\varphi} d\varphi$$

$$f_j(x, \vartheta_t, \tau, \varphi) = \exp\big(C_j(\tau, \varphi) + D_j(\tau, \varphi)\vartheta_t + i\varphi x\big)$$

$$C_j = \big(r_d - r_f\big)\varphi i\tau + \frac{k\theta}{\sigma^2}\left\{(b_1 - \rho\sigma\varphi i + d) - 2\log\left(\frac{1 - g_j e^{d_j t}}{1 - g_j}\right)\right\}$$

$$D_j(\tau, \varphi) = \frac{b_j - \rho\sigma\varphi i + d_j}{\sigma^2}\left(\frac{1 - e^{d_j t}}{1 - g_j e^{d_j t}}\right)$$

$$g_j = \frac{b_j - \rho\sigma\varphi i + d_j}{b_j - \rho\sigma\varphi i - d_j}$$

$$d_j = \sqrt{\big(\rho\sigma\varphi i - b_j\big)^2 - \sigma^2\big(2u_j\varphi i - \varphi^2\big)}$$

$$b_1 = \kappa + \lambda - \sigma\rho$$

$$b_2 = \kappa + \lambda$$

The following are the constraints on the parameters:

$$-1 < \rho < 1$$

$$\kappa > 0$$

$$0 < \theta < 1$$

$$0 < \sigma < 1$$

$$2\kappa\theta > \sigma^2$$

V. Methodology

There are five parameters $(\rho, \kappa, \theta, \sigma, \vartheta_t)$ which need to be estimated in the Heston Model. We do this by observing the real market data, and minimizing the sum of squared errors between the Heston model prices and the real time data. To elaborate, we calculate the heston model prices at the 25D call and put, the 10D call and put and at-the-money. From our observation, we know the actual market prices at these five points. We calculate the sum of the squared errors at these 5 points and choose the set of parameters which minimizes this sum.

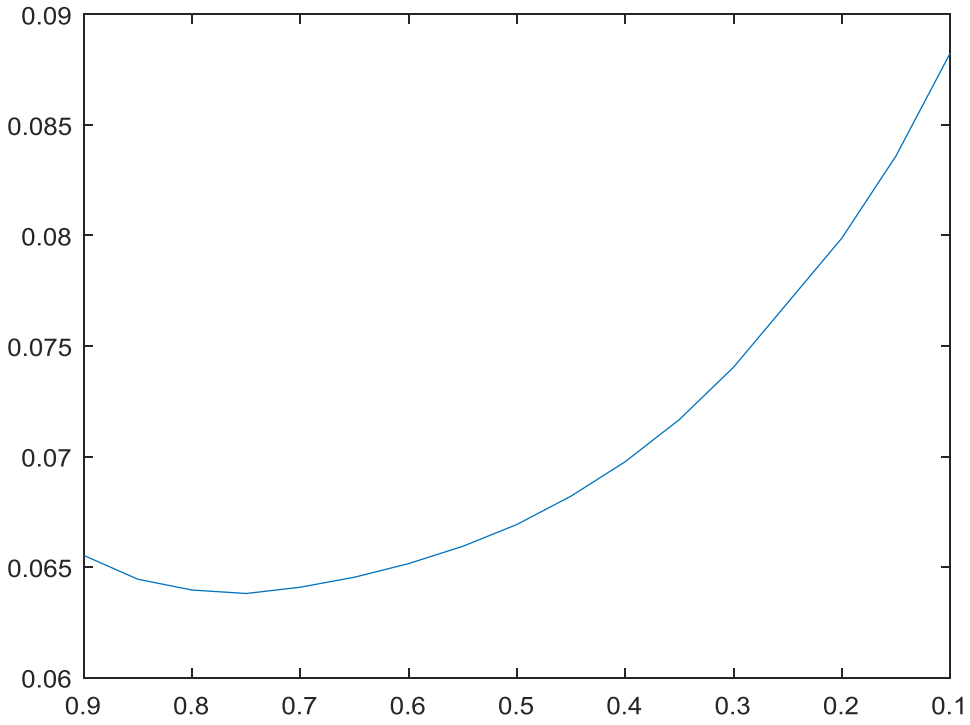
$$Min\ SSE(\rho, \kappa, \theta, \sigma, \vartheta_t) = \sum_{i=1}^{n} (H(\rho, \kappa, \theta, \sigma, \vartheta_t) - Market\ Price\ Of\ Option)^2$$

We use MATLAB for the implementation of the Heston Model. The data being used for calibration and testing is from the Bloomberg terminal at the end of day on 27th Nov, 2015; the currency is USDINR and the 2M,3M,6M,12M maturity volatility curves have been analyzed. In the following graphs, the values on the X axis are the values of the call option deltas (for the purposes of this working paper, the put option delta at any given strike and maturity is simply the volatility of the call option minus 1), the values on the Y axis are the volatilities.

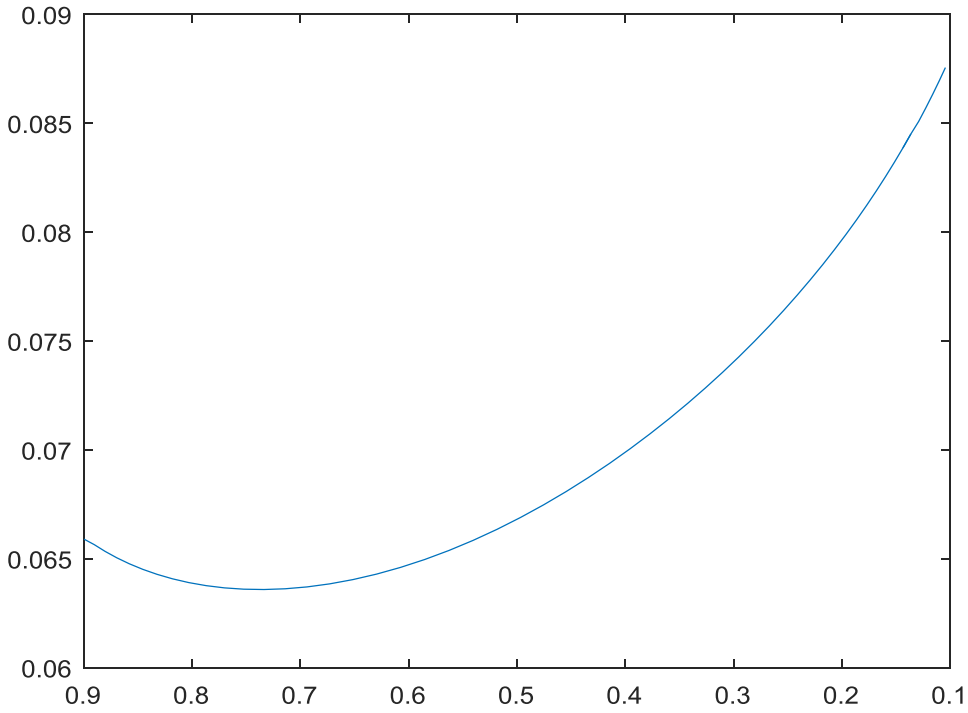
VI. Results

2M Maturity

ObservedVol Curve



Model Volatility Curve

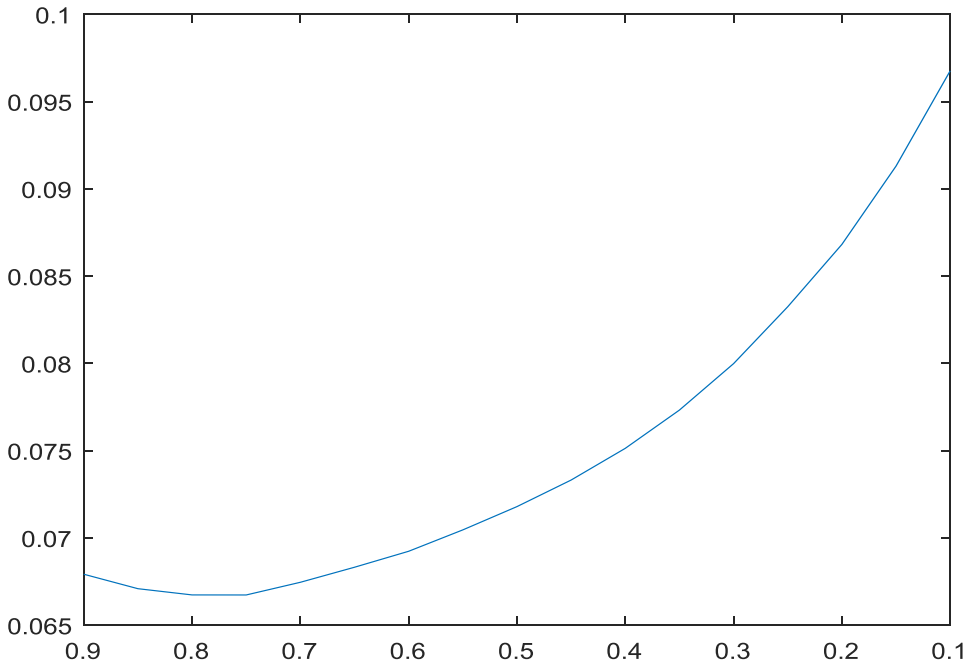


	Market Vols	Model Vols
10P	6.55	6.59
15P	6.44	6.47
20P	6.40	6.41
25P	6.38	6.38
30P	6.41	6.38
35P	6.45	6.41
40P	6.52	6.47
45P	6.59	6.56
ATM	6.69	6.67
45C	6.82	6.81
40C	6.98	6.98
35C	7.17	7.17
30C	7.40	7.40
25C	7.70	7.67
20C	7.99	7.99
15C	8.36	8.36
10C	8.82	8.81

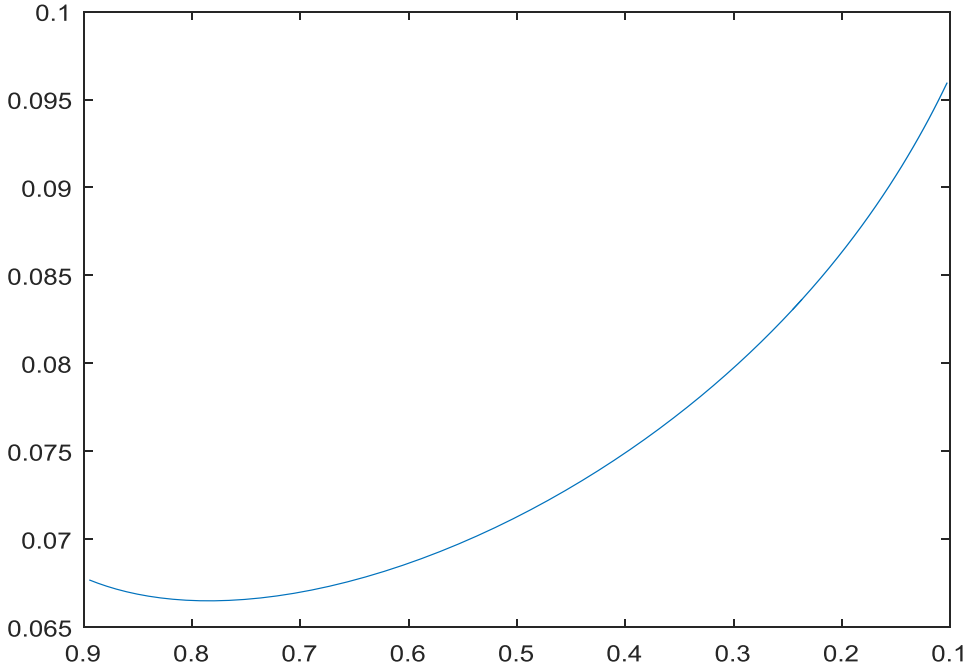
Mean Absolute % Error=0.279%

3M Maturity:

ObservedVol Curve:



Model Vol Curve:

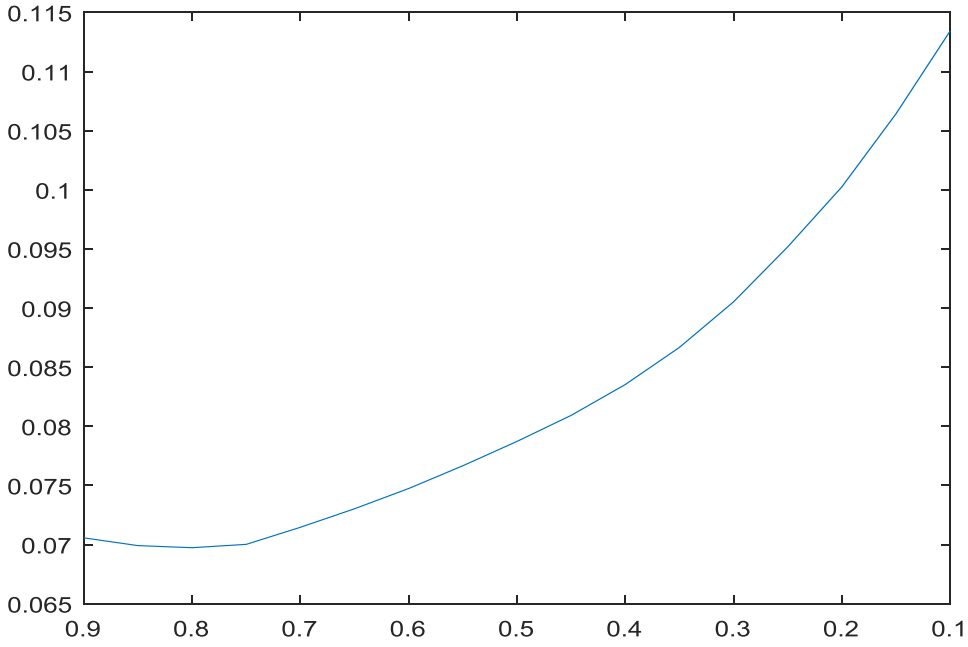


	Market Vols	Model Vols
10P	6.79	6.77
15P	6.71	6.69
20P	6.67	6.65
25P	6.67	6.66
30P	6.75	6.7
35P	6.83	6.77
40P	6.92	6.86
45P	7.05	6.98
ATM	7.18	7.13
45C	7.33	7.3
40C	7.51	7.49
35C	7.73	7.72
30C	8.00	7.98
25C	8.32	8.28
20C	8.68	8.63
15C	9.13	9.07
10C	9.68	9.63

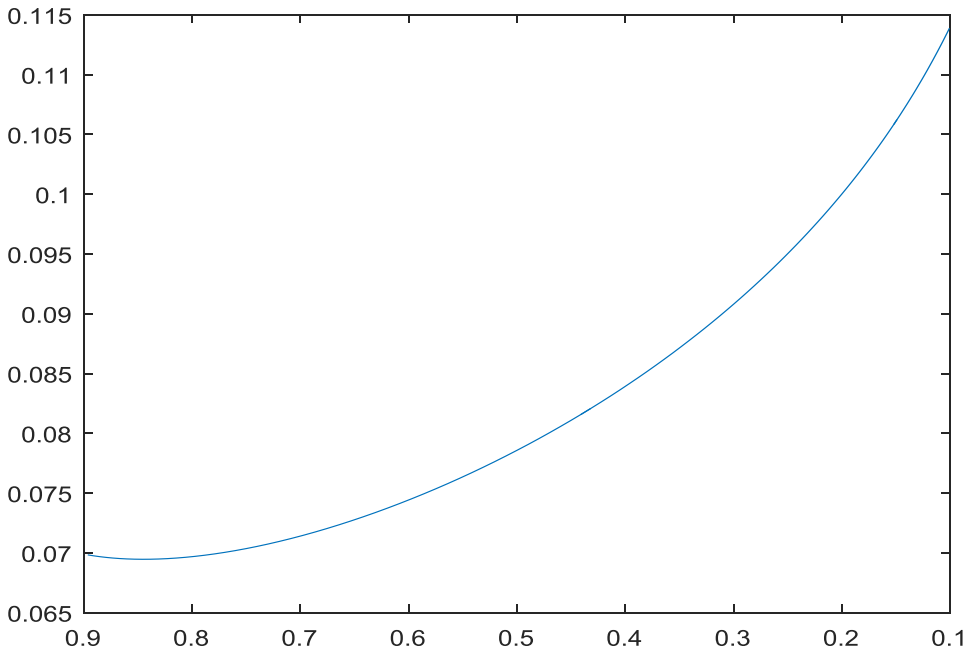
Mean Absolute Percentage Error=0.502%

6M Maturity:

Observed Vol Curve:



Model Vol Curve:

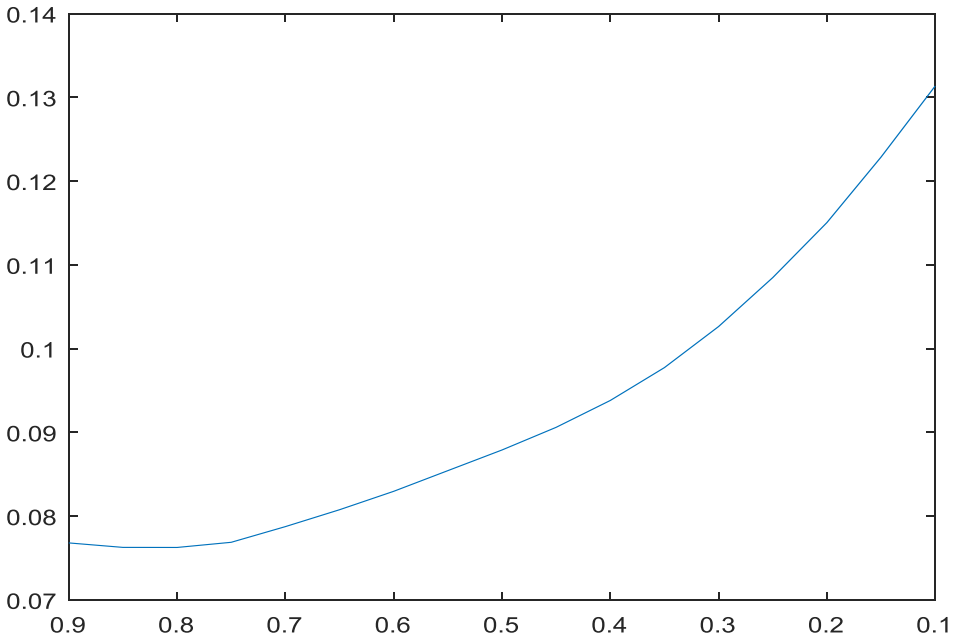


	Market Vols	Model Vols
0.9	7.06	6.99
0.85	6.99	6.95
0.8	6.97	6.97
0.75	7.00	7.04
0.7	7.15	7.14
0.65	7.30	7.28
0.6	7.47	7.44
0.55	7.67	7.64
0.5	7.87	7.86
0.45	8.09	8.11
0.4	8.35	8.39
0.35	8.67	8.71
0.3	9.05	9.08
0.25	9.52	9.50
0.2	10.02	10.00
0.15	10.64	10.61
0.1	11.34	11.40

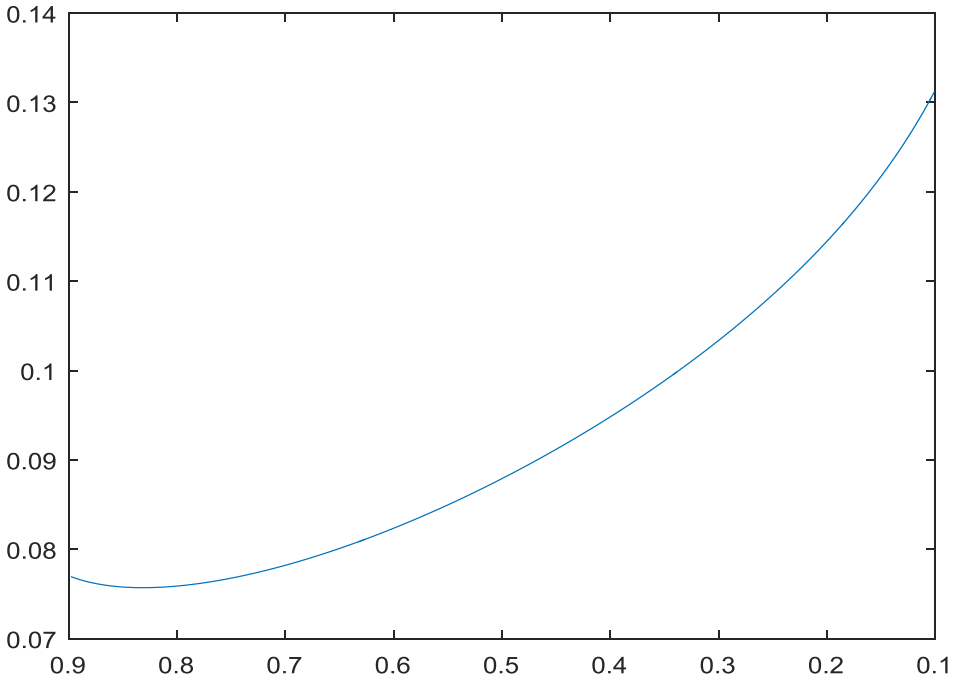
Mean Absolute Percentage Error=0.3637%

12M Maturity:

Observed Vol Curve:



Model Vol Curve:



	Market Vols	Model Vols
10P	7.68	7.70
15P	7.63	7.58
20P	7.63	7.59
25P	7.69	7.68
30P	7.88	7.82
35P	8.08	8.01
40P	8.30	8.24
45P	8.54	8.50
ATM	8.79	8.79
45C	9.06	9.12
40C	9.38	9.48
35C	9.77	9.89
30C	10.26	10.34
25C	10.85	10.85
20C	11.51	11.45
15C	12.29	12.17
10C	13.14	13.14

Mean Absolute Percentage Error=0.55%

VII. Summary

In this working paper, I presented my implementation of the Heston model for the purpose of generating the volatility curve of the USDINR FX market. I first reviewed the FX market and commonly used quoting conventions within the FX market. The Garman Kohlhagen Model and the Heston Model were then introduced and elaborated upon. I then presented the methodology of the calibration and the results of the simulation.

The Mean Absolute Percentage Errors across different deltas for the 4 maturities ranged between 0.3% and 0.5%. The shape and structure of the volatility curves generated by the model are very close to those of the actual implied volatility curves observed in the market. This can partly be attributed to the fact that we used both the 25D as well as the 10D RR,BF for the purposes of calibration; if we had used only the 25D RR,BF it is fairly reasonable to assume that the fit would have been lower. However, it is clear that the Heston model volatility curve matches the observed volatility curve very well and clearly does a much better job than the Black Scholes Model which assumes constant volatility for all strikes.

VIII. References

1) JanekAgnieszka, Kluge Tino,WeronRafal, WystupUwe Peter."FX smile in the Heston model" Statistical Tools for Finance and Insurance, 2011.

2)Crisostomo, Ricardo. "An Analysis of the Heston Stochastic Volatility Model: Implementation and Calibration using Matlab."ComisiónNacionaldel Mercado de Valores(CNMV) Working Paper, 2014.

3) AndersPersson. "Calibration of FX Options and pricing of barrier options" Master's Thesis at Lund University, 2013

4)Yu Tian. "The Hybrid Stochastic-Local Volatility Model with Applications in Pricing FX Options " Doctoral Thesis at Monash University, Australia, 2013

5) Yuan Yang. "Valuing a European option with the Heston model" Master's Thesis at Rochester Institute of Technology, 2013

IX. Appendix: This is the MATLAB code used for implementing the Heston Model for the USDINR market

```
functionbuildsmile(atm,rr25,bf25,rr10,bf10,s,rd,rf,t)


    c10=atm+bf10+rr10/2;
    p10=atm+bf10-rr10/2;
    c25=atm+bf25+rr25/2;
    p25=atm+bf25-rr25/2;
atmk= s.*exp(-1.*norminv(0.5.*exp(rf*t),0,1).*atm.*sqrt(t)+(rd-
rf+0.5.*atm.^2).*t);
test= s.*exp((rd-rf+0.5.*atm.^2).*t);
    ck25= s.*exp(-1.*norminv(0.25.*exp(rf*t),0,1).*c25.*sqrt(t)+(rd-
rf+0.5.*c25.^2).*t);
    ck10= s.*exp(-1.*norminv(0.1.*exp(rf*t),0,1).*c10.*sqrt(t)+(rd-
rf+0.5.*c10.^2).*t);
    pk25= s.*exp(1.*norminv(0.25.*exp(rf*t),0,1).*p25.*sqrt(t)+(rd-
rf+0.5.*p25.^2).*t);
    pk10= s.*exp(1.*norminv(0.1.*exp(rf*t),0,1).*p10.*sqrt(t)+(rd-
rf+0.5.*p10.^2).*t);
atmgk=fxoptioncall(s,atmk,rd,rf,t,atm);
    cgk25=fxoptioncall(s,ck25,rd,rf,t,c25);
    cgk10=fxoptioncall(s,ck10,rd,rf,t,c10);
    pgk25=fxoptionput(s,pk25,rd,rf,t,p25);
    pgk10=fxoptionput(s,pk10,rd,rf,t,p10);
    heston0=[0.1 0.1 0.1 0.1 0.1];
ub=[0.9999 0.9999 0.9999 0.9999 0.9999];
lb=[0.0001 -0.9999 0.0001 0.0001 0.001];
nonlcon_heston=@(heston) nonlcon(heston);
opts = optimoptions('fmincon','Algorithm','sqp');

sse(heston0,s,rd,rf,t,atmk,ck25,ck10,pk25,pk10,atmgk,cgk25,cgk10,pgk25,pgk10)
problem = createOptimProblem('fmincon','objective', ...
```

```
    @(heston)
sse(heston,s,rd,rf,t,atmk,ck25,ck10,pk25,pk10,atmgk,cgk25,cgk10,pgk25,pgk10),
'x0',heston0,'lb',lb,'ub',ub, ...
'options',opts);
gs= GlobalSearch;
    [hest ans1]= run(gs,problem);
    ans1;
    p=1;vol=0;
%calculate smile for calls
for i=50:0.1:80

smilecalls(p)=hestonfxcall(s,hest(1),i,rf,rd,t,hest(2),hest(3),hest(4),hest(5
));
symsx;
        smilecallvols(p)=fzero(@(x)(s.*exp(-rf*t).*normcdf((log(s/i)+(rd-
rf+x^2/2).*t)./(x.*sqrt(t)))-i.*exp(-rd*t).*normcdf((log(s/i)+(rd-rf-
x^2/2).*t)./(x.*sqrt(t)))-smilecalls(p)),0.06);
vol=smilecallvols(p);
        smilecalldeltas(p)=exp(-rf.*t).*normcdf((log(s/i)+(rd-
rf+vol^2/2)*t)./(vol.*sqrt(t)));
        p=p+1;
end
    i=1;
    j=1;
for i=1:length(smilecalldeltas)
if(smilecalldeltas(i)>=0.1 &&smilecalldeltas(i)<=0.9)
deltas(j)=smilecalldeltas(i);
vols(j)=smilecallvols(i);
          j=j+1;
end
end
plot(deltas,vols)
set(gca,'XDir','reverse');

    y90=interp1(deltas,vols,0.90)
    y85=interp1(deltas,vols,0.85)
    y80=interp1(deltas,vols,0.80)
    y75=interp1(deltas,vols,0.75)
    y70=interp1(deltas,vols,0.70)
    y65=interp1(deltas,vols,0.65)
    y60=interp1(deltas,vols,0.60)
    y55=interp1(deltas,vols,0.55)
    y50=interp1(deltas,vols,0.50)
    y45=interp1(deltas,vols,0.45)
    y40=interp1(deltas,vols,0.40)
    y35=interp1(deltas,vols,0.35)
    y30=interp1(deltas,vols,0.30)
    y25=interp1(deltas,vols,0.25)
    y20=interp1(deltas,vols,0.20)
    y15=interp1(deltas,vols,0.15)
    y10=interp1(deltas,vols,0.10)

end
```

```
function h=hestonfxcall(s,v,k,rf,rd,t,rho,sigma,kappa,theta)
    integral1=@(s,v,k,rf,rd,t,theta,kappa,sigma,rho,w) real(exp(-
1i.*w.*log(k)).*hestonfx1(s,v,k,rf,rd,t,rho,sigma,kappa,theta,w)./(1i.*w));
    integral2=@(s,v,k,rf,rd,t,theta,kappa,sigma,rho,w) real(exp(-
1i.*w.*log(k)).*hestonfx2(s,v,k,rf,rd,t,rho,sigma,kappa,theta,w)./(1i.*w));

z1=integral(@(w)integral1(s,v,k,rf,rd,t,theta,kappa,sigma,rho,w),0,500,'RelTo
l',1e-5,'AbsTol',1e-5);

z2=integral(@(w)integral2(s,v,k,rf,rd,t,theta,kappa,sigma,rho,w),0,500,'RelTo
l',1e-5,'AbsTol',1e-5);
    P1=0.5+z1./pi;
    P2=0.5+z2./pi;
    P3=P1;
    P4=P2;
    h=exp(-rf.*t).*s.*P3-k.*exp(-rd.*t).*P4;

end

function h=hestonfxput(s,v,k,rf,rd,t,rho,sigma,kappa,theta)
    integral1=@(s,v,k,rf,rd,t,theta,kappa,sigma,rho,w) real(exp(-
1i.*w.*log(k)).*hestonfx1(s,v,k,rf,rd,t,rho,sigma,kappa,theta,w)./(1i.*w));
    integral2=@(s,v,k,rf,rd,t,theta,kappa,sigma,rho,w) real(exp(-
1i.*w.*log(k)).*hestonfx2(s,v,k,rf,rd,t,rho,sigma,kappa,theta,w)./(1i.*w));

z1=integral(@(w)integral1(s,v,k,rf,rd,t,theta,kappa,sigma,rho,w),0,500,'RelTo
l',1e-5,'AbsTol',1e-5);

z2=integral(@(w)integral2(s,v,k,rf,rd,t,theta,kappa,sigma,rho,w),0,500,'RelTo
l',1e-5,'AbsTol',1e-5);
    P1=0.5+z1./pi;
    P2=0.5+z2./pi;
    P3=1-P1;
    P4=1-P2;
    h=-1.*(exp(-rf.*t).*s.*P3-k.*exp(-rd.*t).*P4);

end

function f1=hestonfx1(s,v,k,rf,rd,t,rho,sigma,kappa,theta,w)
    b1=kappa-sigma.*rho;
    b2=kappa;
    u1=0.5;
    u2=-0.5;
    x=log(s);
    y=log(k);
    d1=sqrt((1i*rho.*sigma.*w-b1).^2-(sigma.^2).*(2*u1.*w.*1i-w.^2));
    d2=sqrt((1i*rho.*sigma.*w-b2).^2-(sigma.^2).*(2*u2.*w.*1i-w.^2));
    g1=(b1-rho.*sigma.*w.*1i+d1)./(b1-rho.*sigma.*w.*1i-d1);
    g2=(b2-rho.*sigma.*w.*1i+d2)./(b2-rho.*sigma.*w.*1i-d2);
    c1=(rd-rf).*w.*1i.*t+kappa.*theta.*((b1-rho.*sigma.*w.*1i+d1).*t-
2.*log((1-g1.*exp(d1.*t))./(1-g1)))./(sigma.^2);
    c2=(rd-rf).*w.*1i.*t+kappa.*theta.*((b2-rho.*sigma.*w.*1i+d2).*t-
2.*log((1-g2.*exp(d2.*t))./(1-g2)))./(sigma.^2);
    D1=(b1-rho.*sigma.*w.*1i+d1).*((1-exp(d1.*t))./(1-
g1.*exp(d1.*t)))./(sigma.^2);
```

```
    D2=(b2-rho.*sigma.*w.*1i+d2).*((1-exp(d2.*t))./(1-
g1.*exp(d2.*t)))./(sigma.^2);
    f1=exp(c1+D1.*v+1i.*w.*x);
    f2=exp(c2+D2.*v+1i.*w.*x);

end

function f2=hestonfx2(s,v,k,rf,rd,t,rho,sigma,kappa,theta,w)
    b1=kappa-sigma.*rho;
    b2=kappa;
    u1=0.5;
    u2=-0.5;
    x=log(s);
    y=log(k);
    d1=sqrt((rho.*sigma.*w.*1i-b1).^2-(sigma.^2).*(2.*u1.*w.*1i-w.^2));
    d2=sqrt((rho.*sigma.*w.*1i-b2).^2-(sigma.^2).*(2.*u2.*w.*1i-w.^2));
    g1=(b1-rho.*sigma.*w.*1i+d1)./(b1-rho.*sigma.*w.*1i-d1);
    g2=(b2-rho.*sigma.*w.*1i+d2)./(b2-rho.*sigma.*w.*1i-d2);
    c1=(rd-rf).*w.*1i.*t+kappa.*theta.*((b1-rho.*sigma.*w.*1i+d1).*t-
2.*log((1-g1.*exp(d1.*t))./(1-g1)))./(sigma.^2);
    c2=(rd-rf).*w.*1i.*t+kappa.*theta.*((b2-rho.*sigma.*w.*1i+d2).*t-
2.*log((1-g2.*exp(d2.*t))./(1-g2)))./(sigma.^2);
    D1=(b1-rho.*sigma.*w.*1i+d1).*((1-exp(d1.*t))./(1-
g1.*exp(d1.*t)))./(sigma.^2);
    D2=(b2-rho.*sigma.*w.*1i+d2).*((1-exp(d2.*t))./(1-
g1.*exp(d2.*t)))./(sigma.^2);
    f1=exp(c1+D1.*v+1i.*w.*x);
    f2=exp(c2+D2.*v+1i.*w.*x);

end

function
y=sse(heston,s,rd,rf,t,atmk,ck25,ck10,pk25,pk10,atmgk,cgk25,cgk10,pgk25,pgk10
)

    v=heston(1);
rho=heston(2);
sigma=heston(3);
kappa=heston(4);
theta=heston(5);
atmhc=hestonfxcall(s,v,atmk,rf,rd,t,rho,sigma,kappa,theta);
    hc25=hestonfxcall(s,v,ck25,rf,rd,t,rho,sigma,kappa,theta);
    hc10=hestonfxcall(s,v,ck10,rf,rd,t,rho,sigma,kappa,theta);
    hp25=hestonfxput(s,v,pk25,rf,rd,t,rho,sigma,kappa,theta);
    hp10=hestonfxput(s,v,pk10,rf,rd,t,rho,sigma,kappa,theta);
    y=sqrt((atmhc-atmgk)^2+(hc25-cgk25)^2+(hc10-cgk10)^2+(hp25-
pgk25)^2+(hp10-pgk10)^2);


end
```