A NOTE ON THE PRICING OF USD SWAPTIONS UNDER THE SABR LMM MODEL

CHRISTIAN CRISPOLDI

First Version: 13 December 2015

Contents

1. Introduction

In this brief note we will price USD swaptions using the SABR LMM model introduced in Chapter 7 of our book "SABR and SABR LIBOR Market Models in Practice: With Examples Implemented in Python." The idea is to broaden the real market data examples already provided in our book, expanding to the USD market. The data is as of April, 30th 2015.

2. USD Swaption Pricing Under the SABR LMM Model With Null Forward-Volatility Correlation

In [1] we have been introducing and working with a null forward-volatility correlation matrix SABR LMM. In particular, all the elements $\phi_{k,l}$ of the matrix Π, which is defined as follows

$$\Pi = \left[\begin{array}{cc} \rho & \phi \\ \phi^{\dagger} & \vartheta \end{array} \right], \tag{2.1}$$

have been set to zero. This facilitates the model set-up, as well as calibration and simulation performances (cf. [1], Section 7.4). A drawback of this method is that when working in markets characterized by high $\beta_{m,n}$, the Rebonato and White swaption approximations (required to calibrate to the swaption market) perform poorly (more on this later). However, as the USD market is not currently represented by an high $\beta_{m,n}$, we can take full advantage of the null forward-volatility SABR LMM model set-up.

The USD swaption market has been calibrated through the Rebonato and White approximations (cf. [1], Section 7.5.3) where, instead of using the usual $g\left(T_{k-1}-t\right)$ and $h\left(T_{k-1}-t\right)$ parametric forms, we have been employing piece-wise constant α_k and ν_k (we have been effectively using two bi-dimensional matrices, one for each of the two parameters). Those matrices have been obtained through an algorithm inspired by the one propesed by Brigo and Morini [2] for the standard LMM. The advantage of employing piece-wise constant α_k and ν_k over $g\left(T_{k-1}-t\right)$ and $h\left(T_{k-1}-t\right)$ is the added flexibility required when calibrating large portions of the swaption volatility matrix.

To further enhance the simulation performances we have been using Rebonato's weight freezing technique to compute the forward swap rate $S_{m,n}\left(t\right)$ (cf. [1], Section 6.4.2). The SABR LMM simulation has been performed using the code presented in [1], Table 7.7, slightly modified to accomodate for the matrices of piece-wise constant α_k and ν_k (instead of using g_t, h_t) and to be able to take advantage of the weight freezing technique. The results for a $5y2y$ and a $10y5y$ swaption obtained from the aforementioned SABR LMM Monte Carlo code are reported in Figures 2.1 and 2.2. The simulation has been performed with $n_{sim} = 100000$ and four time steps per year. We have compared the SABR LMM results, with those of a SABR Monte Carlo in which the swaption $\alpha_{m,n}, \beta_{m,n}, \nu_{m,n}, \rho_{m,n}$ have been directly used in the simulation. As can be seen the agreement between the two methods is outstanding. This coupled with the extremely fast computation which the weight freezing technique and a four time steps per year time partition bring, allows us to have an extremely perfomant model.

As mentioned, the Rebonato and White swaption approximations perform poorly when $\beta_{m,n}$ approaches 1. This can be easily showed by taking the $5y2y$ SABR parameters, and changing $\beta_{m,n}$ to 0.9. The results are displayed in Figure 2.3.

For high $\beta_{m,n}$ markets we need therefore to resort to a SABR LMM model where $\phi_{k,l}$ are not fixed to zero and the β_k are set *a priori*.

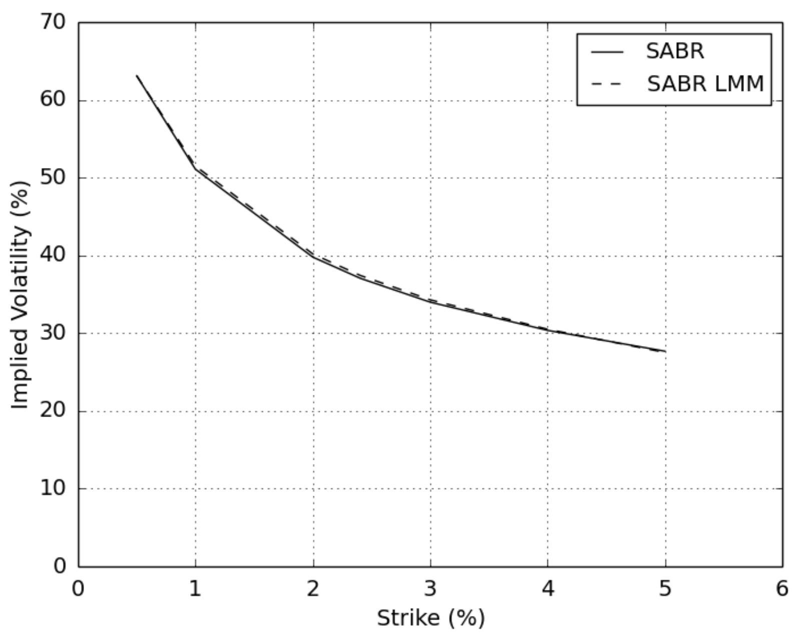

FIGURE 2.1. Black implied volatilities produced by SABR and SABR LMM for a $5y2y$ USD swaption. Both SABR and SABR LMM have been simulated using a Monte Carlo Euler scheme (cf. [1], Table 5.5 and Table 7.7 for the Python code) with $n_{sim} = 10^5$ and $n_{step} = 20$. The swaption SABR parameters used are: $\alpha_{m,n} = 0.02155, \beta_{m,n} = 0.23568, \nu_{m,n} = 0.19565, \rho_{m,n} = 0$ and $S_{m,n} = 2.407\%$.

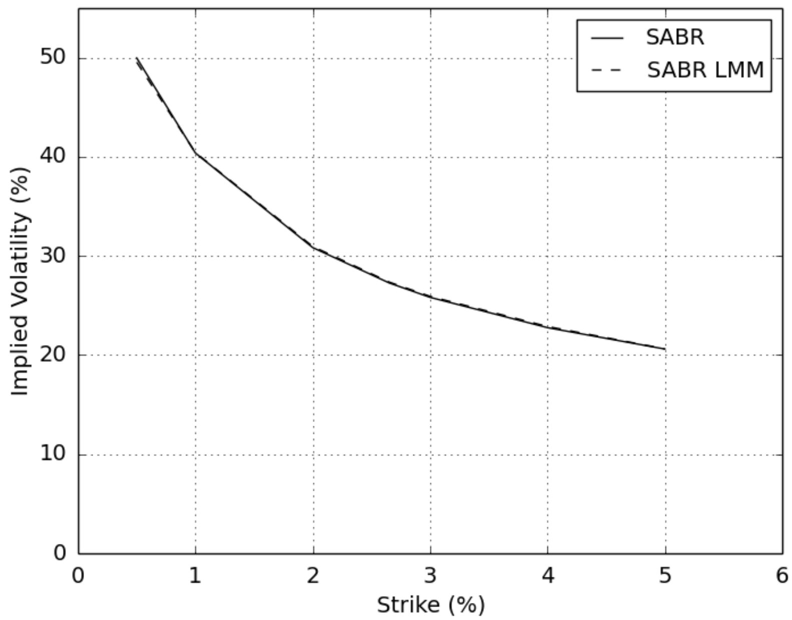

FIGURE 2.2. Black implied volatilities produced by SABR and SABR LMM for a $10y5y$ USD swaption. Both SABR and SABR LMM have been simulated using a Monte Carlo Euler scheme (cf. [1], Table 5.5 and Table 7.7 for the Python code) with $n_{sim} = 10^5$ and $n_{step} = 40$. The swaption SABR parameters used are: $\alpha_{m,n} = 0.01012, \beta_{m,n} = 0.08071, \nu_{m,n} = 0.23092, \rho_{m,n} = 0$ and $S_{m,n} = 2.6514\%$.

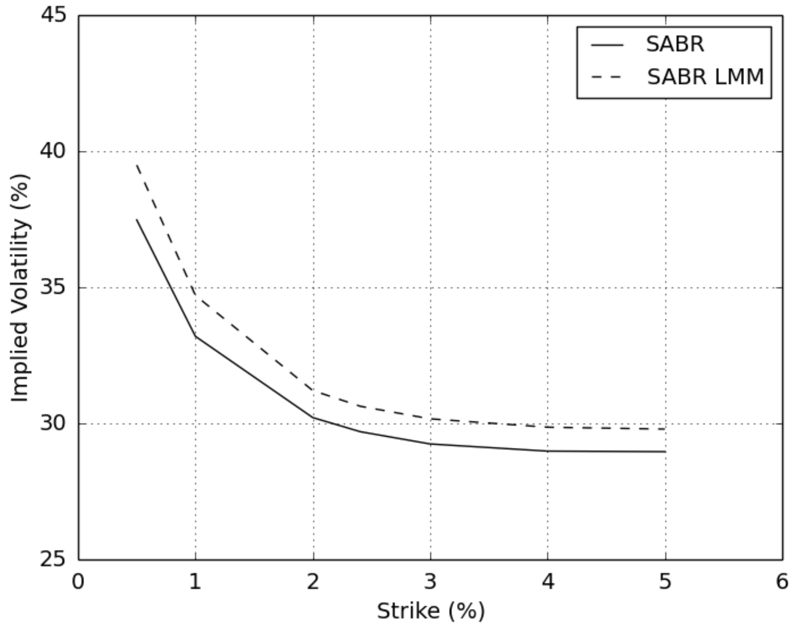

FIGURE 2.3. Black implied volatilities produced by SABR and SABR LMM for a $5y2y$ USD swaption. Both SABR and SABR LMM have been simulated using a Monte Carlo Euler scheme (cf. [1], Table 5.5 and Table 7.7 for the Python code) with $n_{sim} = 10^5$ and $n_{step} = 20$. The swaption SABR parameters used are: $\alpha_{m,n} = 0.02155, \beta_{m,n} = 0.9, \nu_{m,n} = 0.19565, \rho_{m,n} = 0$ and $S_{m,n} = 2.407\%$.

REFERENCES

[1] Crispoldi, C., Wigger, G., and Larkin, P., SABR and SABR LIBOR Market Models in Practice, With Examples Implemented in Python, 1st Edition, Palgrave Macmillan, 2015

[2] Brigo, D., and Morini, M., An empirically efficient analytical cascade calibration of the LIBOR Market Model based only on directly quoted swaptions data, Working Paper, 2005