Reghai Adil,
Equity and Commodity Markets,
NXS Quantitative at Natixis,
adil.reghai@natixis.com

Improve your Calibration thanks to AAD and regularisation

December 30, 2021

Abstract

In this paper we show how to adjust your local volatility surface in order to improve your calibration. It starts by computing the distance to the market and producing local volatility sensitivities w.r.t. local volatility points using AAD. We then apply constrained optimisation algorithm in order to adjust the numerical model to the market data.

1 Introduction

There is always a slight discrepancy between the theoretical formula and the numerical instance that is implemented to produce prices and hedges. For example, if one implements the Dupire formula [1] using analytical approach. Then he produces prices using his favorite numerical technique (Monte Carlo or PDE) he will end up with an error : few basis points up to few percent if a succession of errors add up. In this paper we take the error as part of the numerical process, measure them on vanilla options and show how to adjust the model in order to erase these errors.

2 Formulation of the problem

Notations

We introduce the open loop local volatility $(t, S) \rightarrow \sigma(t, S)$,

We introduce V_p the vanilla option of maturity T_p,

We note π_p^{mkt} the market price of the vanilla options for $p = 1, \cdots, c$

We note $\pi_p^{\sigma} = \mathbb{E}_{\sigma(t,S)}[V_p(T_p)]$ for the European option for $p = 1, \cdots, c$

Our objective is to compute an adjustment to the the local volatility $(t, S) \rightarrow \tilde{\sigma}(t, S)$ in order to improve the fit.

Initial Local volatility

We show in this step how to compute the initial local volatility surface. The formula can be found in [2]. We build a time grid $t_i = i\delta_t$ for $i \in [0, n_t]$ and $X_j = X_0 + j\delta_x$ for $j \in [0, n_x]$.

$$\sigma(i,j)^2 = \frac{\Sigma^2(i,j) + 2i\Sigma(i,j)(\Sigma(i+1,j) - \Sigma(i,j))}{1 + 2d_1\sqrt{i\delta_t}\frac{\Sigma(i,j+1)-\Sigma(i,j-1)}{e^{\delta_x}-e^{-\delta_x}} + d_1 d_2 i\delta_t(\frac{\Sigma(i,j+1)-\Sigma(i,j-1)}{e^{\delta_x}-e^{-\delta_x}})^2 + i\delta_t\Sigma(i,j)\partial_{i,i}\Sigma(i,j)} \tag{1}$$

where,

- $d_1 = \frac{-j\delta_x + \Sigma(i,j)^2 i\delta_t}{\Sigma(i,j)\sqrt{i\delta_t}}$
- $d_2 = d_1 - \Sigma(i,j)\sqrt{i\delta_t}$

- and $\frac{1}{2}\partial_{i,i}\Sigma(i,j)(e^{\delta_x}-1)(1-e^{-\delta_x}) = \frac{e^{\delta_x}-1}{e^{\delta_x}-e^{-\delta_x}}\Sigma(i,j-1) + \frac{1-e^{-\delta_x}}{e^{\delta_x}-e^{-\delta_x}}\Sigma(i,j+1) - \Sigma(i,j)$

For a given time slice i, the previous formula can fail either because it is not positive or because of a division par zero. In either cases, the corresponding j points should not be computed. One should regenerate the missing points by interpolating continuously using the valid points.

Preliminary Work

and the sensitivities for all vanillas with respect to local volatility points at the cost of only one Monte Carlo simulation.

- Prepare the **data structure** of results, This consists of p matrices of different sizes depending on the maturity of the product. To clarify the notations we shall assume that the p vanilla products have maturities $T_1 \leq T_2 \ldots \leq T_p$ corresponding to indices $i_1 \leq i_2 \ldots \leq i_p$. Meaning that $t_{i_q} = T_q$. Therefore we allocate a matrix of size $n_x \times i_q$ for the product with maturity T_q.
- **Forward sweep** for one trajectory,

```
    initialise
        $X(0) = 0$
loop over time
        $\sigma(i,X(i)) = (1-p_i)\sigma(i,j) + p_i\sigma(i,j+1)$
        where
            $j = [\frac{X(i)-X_0}{\delta_x}]$
            $p_i = \frac{X(i)-X(j)}{X(j+1)-X(j)}$
            $d_x\sigma_i = \partial_x\sigma(i,X(i))$
        $X(i+1) = X(i) - \frac{1}{2}\sigma(i,X(i))^2(t_{i+1}-t_i) + \sigma(i,X(i))(W(t_{i+1})-W(t_i))$
```

- **Backward sweep**,

```
    initialise
        $\bar{X}(n_t) = 1$
loop over time
        $\bar{X}(i) = \bar{X}(i+1)(1 + d_x\sigma_i(W(t_{i+1}) - W(t_i) - \sigma(i,X(i))(t_{i+1}-t_i))$
        $\bar{\sigma}(i,X(i)) = \bar{X}(i+1)(W(t_{i+1}) - W(t_i) - \sigma(i,X(i))(t_{i+1}-t_i)$
        $\bar{\sigma}(i,j) = (1-p_i)\bar{\sigma}(i,X(i))$
        $\bar{\sigma}(i,j+1) = p_i\bar{\sigma}(i,X(i))$
```

- **Populating the sensitivities for all vanilla products**,
  for $p = 1 \ldots c$ update the sensitivity matrix
  $\bar{\sigma}_p(i,j) := \frac{\partial\pi_p^\sigma}{\partial\sigma(t_i,S_j)} + = \frac{1}{N} 1_{\frac{\{e^{X_{T_p}} \geq K_p\}}{\bar{X}(i_p)}} \bar{\sigma}(i,j)$
  $\bar{\sigma}_p(i,j+1) := \frac{\partial\pi_p^\sigma}{\partial\sigma(t_i,S_{j+1})} + = \frac{1}{N} 1_{\frac{\{e^{X_{T_p}} \geq K_p\}}{\bar{X}(i_p)}} \bar{\sigma}(i,j+1)$

Inputs

We compute the mismatch $\delta\pi_1$, ..., $\delta\pi_c$ for our c vanilla options,

$$\delta\pi_p \quad := \quad \pi_p^{mkt} - \pi_p^\sigma \tag{2}$$

In theory, mismatch should be zero. In practice, it is not the case, because of computational , convergence and discretization errors.

We approximate the price variation thanks to first order expansion thanks to the sensitivities $\frac{\partial \pi_p^\sigma}{\partial \sigma(t,S)}$,

$$\delta\pi_p \quad \approx \quad \int_0^T \int_0^\infty \frac{\partial \pi_p^\sigma}{\partial \sigma(t,S)}(\tilde{\sigma}(t,S) - \sigma(t,S))dSdt \tag{3}$$

Output

We produce a local vol matrix $\delta\sigma(t,S)$ defined as the difference between $\tilde{\sigma}(t,S)$ and $\sigma(t,S)$.

$$\delta\sigma(t,S) := \tilde{\sigma}(t,S) - \sigma(t,S)$$

In practice, we display in the system a finite-dimensional matrix for a set of spots and times $(t_i, S_j)_{1 \le i \le N, 1 \le j \le M}$.

Our objective is to construct $\delta\sigma(t_i, S_j)$ such that

$$\sum_{i,j} \frac{\partial \pi_p^\sigma}{\partial \sigma(t_i,S_j)} \delta\sigma(t_i,S_j) \quad = \quad \delta\pi_p, \quad p = 1, \cdots, c$$

For a finite number of products, this condition does not enable to identify the matrix $\delta\sigma(t_i, S_j)$. As a solution, we will also impose that we are close to the initial local volatility, therefore we assume that the norm of $\delta\sigma(t_i, S_j)$ to be small.

Optimization

Writing a matrix $A \in M_{N\times M}$ in its *vectorial representation* $\bar{A} \in \mathbb{R}^{N\times M}$ [1] , and using the Euclidian norm $||A||^2 := \langle \bar{A}, \bar{A} \rangle$, our problem can be formulated as a minimization problem. By using the matrix notations:

$$\begin{aligned} Y_{i,j} &:= \delta\sigma(t_i,S_j) \\ \delta^p_{i,j} &:= \frac{\partial \pi_p^\sigma}{\partial \sigma(t_i,S_j)} \end{aligned}$$

the problem writes:

$$Y^* := \text{argmin}\{Y \text{ s.t. } (5) \ : \ \frac{1}{2}||Y||^2\} \tag{4}$$

$$\langle \bar{Y}, \bar{\delta^p} \rangle = \delta\pi_p, \quad \forall p = 1, ..., c \tag{5}$$

Solution of the optimization

The previous optimization (4) has an analytic formula, a linear combination of the matrices $(\delta^p)_{1 \le p \le c}$:

$$Y^* = -\sum_{p=1}^{c} \lambda_p \delta^p \tag{6}$$

with $(\lambda_1, ..., \lambda_c)^\top = B^{-1}V, \quad B \in M_{c\times c}$ and $B_{pq} = \langle \bar{\delta^p}, \bar{\delta^q} \rangle, \quad V \in \mathbb{R}^c$ and $V_p = -\delta\pi_p$.

We can also solve the previous system using an incremental approach from small maturities to larger ones. This has the advantage to inverse small sized matrices at each step.

[1] $\bar{A}$ is of size $n = N \times M$, with the following mapping : the j^{th} column of A is between the jN and $(j+1)N - 1$ elements of $\bar{A}$.

3 Computing the Vega KT

Once this model has been calibrated, we have created a strong link between the model parameters $\sigma(t, S)$ and the tradeable products represented through $\Sigma(T_i, K_j)$. Indeed, the matrix B contains the information about the space spanned by the calibration products. If we price a new product, we can compute

Now we assume that we price an exotic product under this model and we compute $\frac{\partial \pi^\sigma}{\partial \sigma(t_i,S_j)}$. We compute the vector of size $p \times 1$, $\delta(k) = \sum_{i,j} \frac{\partial \pi^\sigma}{\partial \sigma(t_i,S_j)} \frac{\partial \pi^{\sigma\, k}}{\partial \sigma(t_i,S_j)}$.

The vega KT is obtained as previously as follows:

$$\lambda^\star = B^{-1}\delta \tag{7}$$

4 Numerical Examples

5 Conclusion

In this paper we have shown a method to improve existing open loop calibrations to become closed loop calibration. We applied it to the European local volatility calculation to obtain a model that calibrates perfectly to the vanilla. Also, thanks to the available calculations, one can immediately compute the super vega bucket.

References

[1] B. Dupire. Pricing with a smile.Risk Magazine: 18-20, 1994.

[2] J. Gatheral. The volatility surface, a Practitionner's guide.2006.

[3] Reghai, Adil and Boya, Gilles and Vong, Ghislain, Local Volatility: Smooth Calibration and Fast Usage (February 20, 2012). Available at SSRN: https://ssrn.com/abstract=2008215 or http://dx.doi.org/10.2139/ssrn.2008215

[4] A. Reghai
Quantitative Finance, back to basics.
Palgrave Mac Millan, 2015.

[5] Reghai, Adil and Langnau, Alex and Ben Haj Yedder, Adel, Decoding the American Vanilla Prices (October 31, 2013). Available at SSRN: https://ssrn.com/abstract=2347910 or http://dx.doi.org/10.2139/ssrn.2347910