DEEP PPDES FOR ROUGH LOCAL STOCHASTIC VOLATILITY

ANTOINE JACQUIER AND MUGAD OUMGARI

Abstract. We introduce the notion of rough local stochastic volatility models, extending the classical concept to the case where volatility is driven by some Volterra process. In this setting, we show that the pricing function is the solution to a path-dependent PDE, for which we develop a numerical scheme based on Deep Learning techniques. Numerical simulations suggest that the latter is extremely efficient, and provides a good alternative to classical Monte Carlo simulations.

1. Introduction

Stochastic models in financial modelling have undergone many transformations since the Black and Scholes model [12], and its most recent revolution, pioneered by Gatheral, Jaisson and Rosenbaum [34], has introduced the concept of rough volatility. In this setting, the instantaneous volatility is the solution of a stochastic differential equation driven by a fractional Brownian motion with small (less than a half) Hurst exponent, synonym of low Hölder regularity of the paths. Not only is this feature consistent with historical time series [34], but it further allows to capture the notoriously steep at-the-money skew of Equity options, as highlighted in [3, 4, 8, 29, 30]. Since then, a lot of effort has been devoted to advocating this new class of models and showing the full extent of their capabilities, in particular as good estimators for a large class of assets [11], and for consistent pricing of volatility indices [10, 51]. Nothing comes for free though, and the flip side of this new paradigm is the computational cost. With the notable exception of the rough Heston model [11, 26, 27, 25, 24] and its affine extensions [2, 35], the absence of Markovianity of the fractional Brownian motion prevents any pricing tools other than Monte Carlo simulations; the simulation of continuous Gaussian processes, including fractional Brownian motion, is traditionally slow as soon as one steps away from the standard Brownian motion. However, the clear superiority–for estimation and calibration–of these rough volatility models has encouraged deep and fast innovations in numerical methods for pricing, in particular the now standard Hybrid scheme [10, 43] as well as Donsker-type theorems [45, 64], numerical approximations [6, 42, 36] and machine learning-based techniques [47, 71].

In fact, industry practice is often entrenched, not in pure stochastic volatility models, but in models enhanced with a local volatility component, à la Dupire [18], thereby ensuring an exact fit to the observed European option price surface. The natural next step for rough volatility models is therefore to include such a component. This is the main theme of the present work, where we provide, for the first time as it seems, a numerical algorithm able to evaluate options, European or path-dependent, in such a rough local stochastic volatility setting. Although this could be performed by simulation, we adopt a different strategy, following the recent development by Viens and Zhang [72], who proved an analogous version of the Feynman-Kac theorem for rough volatility models. The fundamental difference, though, is that the corresponding partial differential equation is now path-dependent. This forces us to revisit classical market completeness results in the setting of rough local stochastic volatility models. Path-dependent partial differential equations (PPDEs) have been studied extensively by Touzi, Zhang and co-authors [22, 21, 20], and we shall draw existence and uniqueness from their works. Despite these advances, though, very little has been developed to solve these PPDEs numerically, with the sole exception of a path-dependent version of the Barles and Souganidis' monotone scheme [5] by Zhang and Zhuo [73] and Ren and Tan [66]. The implementation thereof is however far from obvious, and we propose instead a novel algorithm, based on the discretisation of the pricing PPDE,

Date: June 6, 2019.

2010 Mathematics Subject Classification. 35R15, 60H30, 91G20, 91G80.

Key words and phrases. rough volatility, Deep learning, Path-dependent PDEs.

The views and opinions expressed here are the authors' and do not represent the opinions of their employers. They are not responsible for any use that may be made of these contents. No part of this presentation is intended to influence investment decisions or promote any product or service. The authors would like to thank Lukasz Szpruch for stimulating discussions.
which we then solve using the deep learning technique pioneered by Jentzen [11], which does not suffer the
curse of dimensionality.

We note in passing that using machine learning (or deep learning) techniques to solve high-dimensional
PDEs has recently been the focus of several approaches. Neural networks have indeed been used to solve
PDEs for a long time [57, 68, 56, 60]; more recently, Sirignano and Spiliopoulos [70] proposed an algorithm
not depending on a given mesh (as opposed to the previous literature), thus allowing for an easier extension
to the multi-dimensional case. During the writing-up of the present paper, a further two ideas, similar in
spirit to the one we are borrowing from [11] came to light: Sabate-Vidales, Šiska and Szpruch [64], as well
as and Huré, Pham and Warin [18] also used the BSDE counterpart of the PDE—albeit in different ways—to
apply machine learning techniques. Since our set-up here is not solely about solving a high-dimensional
PDE, we shall leave the precise comparison of these different schemes to rest for the moment.

We shall follow a natural progression in this paper: Section 2 introduces the financial modelling setup of the
analysis, introducing rough local stochastic volatility models, and proving preliminary results fundamental
for their application in quantitative finance. In Section 4 we show that, in this context, a financial derivative
is the solution to a path-dependent PDE, for which we propose a discretisation algorithm, and adapt the
Deep Learning methodology developed in [11]. We show the validity and accuracy of this technique in
Section 5 for different examples of interest. We gather in appendix some long proofs and reminders in order
not to disrupt the the flow of the paper.

2. Modelling framework

In order to dive right into the modelling framework and our main setup, we postpone to Appendix A a
review of the functional Ito formula for stochastic Volterra systems, as developed by Viens and Zhang [72].
We introduce a rough local stochastic volatility model for the dynamics of a stock price process. Before diving
into numerical considerations, we adapt the classical framework of no-arbitrage and market completeness to
this setup in order to ensure that pricing and calibration make any sense at all.

2.1. Rough local stochastic volatility model. We are interested here in local stochastic volatility models,
where the volatility is rough, in the sense of [13]. This can be written, under the historical measure, as
\begin{equation}
\begin{aligned}
S_t &= S_0 + \int_0^t \mu_r S_r dr + \int_0^t l(r, S_r, V_r) S_r dW_r, \\
V_t &= V_0 + \int_0^t K(t-r) \left( b(V_r) dr + \xi(V_r) dB_r \right),
\end{aligned}
\end{equation}
where \( \rho \in [-1, 0] \) and \( W \) and \( B \) are two standard Brownian motions. Setting \( X = (S, V) \), we can rewrite
the system as
\[
X_t = X_0 + \int_0^t \mathbf{b}(t, r, X_r) dr + \int_0^t \mathbf{\sigma}(t, r, X_r) \cdot dB_r,
\]
where
\[
\mathbf{b}(t, r, X_r) = \begin{pmatrix} \mu_r S_r \\ K(t-r)b(V_r) \end{pmatrix} \quad \text{and} \quad \mathbf{\sigma}(t, r, X_r) = \begin{pmatrix} p l(t, S_r, V_r) S_r & \rho l(t, S_r, V_r) S_r \\ K(t-r) \xi(V_r) & K(t-r) \xi(V_r) \end{pmatrix},
\]
where \( p := \sqrt{1 - \rho^2} \) and \( \mathbf{B} = (B^1, B) \), with \( W := \rho B + p B^1 \). The SDE for \( X \) represents a stochastic Volterra
system which is, in general, not Markovian. We could in principle allow for more generality and assume,
following [72], that for any \( t \geq 0 \) and \( r \in [0, t] \), the coefficients \( \mathbf{b} \) and \( \mathbf{\sigma} \) depend on the past trajectory \( X_{r \\
\wedge t} \), for example to include models with delay. However, such an extension is not needed in the application we
are interested in, and we shall not pursue it. We restrict the correlation parameter \( \rho \) to be non positive
in order to ensure—with some assumptions, below, on the coefficients—that the stock price process \( S \) is a
true martingale. This assumption is not so restrictive in Equity or FX markets, where it is respectively quite negative and close to zero. We assume that the function \( l(\cdot) \) is of the form \( l(t, S, v) = \mathcal{L}(t, S) \varsigma(v) \), where \( \mathcal{L}(\cdot, \cdot) \) is usually called the leverage function. From the results by Dupire [15] and Gyöngy [31], if one
wants to ensure that this model calibrates exactly to European option prices, then the equality
\[
\sigma_L^2(t, s) = \mathbb{E}^Q \left[ l(t, S_t, V_t)^2 | S_t = s \right] = \mathbb{E}^Q \left[ \mathcal{L}(t, S_t)^2 \varsigma(V_t)^2 | S_t = s \right] = \mathcal{L}(t, s)^2 \mathbb{E}^Q \left[ \varsigma(V_t)^2 | S_t = s \right]
\]
must hold for every \( t, s \geq 0 \), where the function \( \sigma_L \) is called the local volatility can be, at least in theory, be obtained directly from European option prices; we refer the reader to Section 3.4 for more details, and for some numerical aspects. Here \( \mathbb{Q} \) denotes the risk-neutral measure. We will show below (Assumption 2.1 and Theorem 4.3) that such a probability measure exists, thereby making the model meaningful for option pricing. The leverage function can then be recovered directly as
\[
\mathcal{L}(t, s) = \frac{\sigma_L(t, s)}{\sqrt{\mathbb{E}^\mathbb{Q}_s [\varsigma(V_t)^2|S_t = s]}},
\]
as long as the right-hand side makes sense, and Assumption 2.1 ensures this is indeed the case. Note that the term on the right-hand side is a conditional expectation with respect to the stock price, and therefore only depends on \( t \) and \( \{S_t = s\} \), no matter whether the variance process is Markovian or not. This class of models has the advantage of ensuring perfect (at least theoretically) calibration to European option prices, while giving flexibility to price other options, in particular path-dependent or exotic options. A full account quantitative finance, such as power-law kernel \( K(\cdot) \), would be of such models and the corresponding calibration problem can be found in [2]. A rigorous proof of the existence and uniqueness of (2) is outside the scope of this paper; in fact, even in the classical (non-rough case), a general answer does not exist yet, and only recently some advances [3, 4] have been made in this direction.

We will develop some numerics for a rough local stochastic volatility model in Section 4.2, but for now leave this concept aside and consider a general function \( \mathcal{L} \). We shall further assume the existence of a money market account yielding a risk-free interest rate \((r_t)_{t \geq 0}\). We shall always work under the following considerations:

**Assumption 2.1.**

1. The kernel \( K \in L^2_{\text{loc}}(\mathbb{R}_+ \rightarrow \mathbb{R}) \) admits a resolvent of the first kind, and there exists \( \gamma \in (0, 2] \) such that
   \[
   \int_0^h K(t)^2 dt = O(h^\gamma) \quad \text{and} \quad \int_0^T [K(t + h) - K(t)] dt = O(h^\gamma), \quad \text{for every } T \geq 0, \text{ as } h \text{ tends to zero};
   \]
2. the functions \( b \) and \( \xi^2 \) are linear of the form \( b(y) = b_0 + b_1 y \) and \( \xi(y)^2 = a_0 + a_1 y \);
3. for any \( t \geq 0 \), the map \( \mathcal{L}(t, \cdot) \) is bounded away from zero and bounded above by \( C_\mathcal{L} \);
4. the system (2) admits a unique (weak) solution;
5. the function \( \varsigma \) is strictly positive, bounded away from zero, uniformly Hölder continuous and \( \varsigma(y) \leq C_\varsigma (1 + |y|^{p_\varsigma}) \) for some \( C_\varsigma > 0 \) and \( p_\varsigma \in (0, 1) \).

The form of the kernel \( K \) ensures that the variance process is stationary. We borrow Condition (ii) from [2] so that, in the purely stochastic volatility case \( \mathcal{L}(\cdot, \cdot) \equiv 1 \), taking \( \varsigma(y) = \sqrt{y} \), we are exactly in the setting of an affine Volterra system \((\log(S), V)\). In fact, this condition alone ensures that the process \( V \) is an affine Volterra process, and by [3, Theorem 3.3], Assumption 2.1(i) ensures that the SDE for \( V \) admits a continuous weak solution, with \( \sup_{t \geq 0} \mathbb{E}[|V_t|^p] \) finite for all \( p \geq 2 \). This in particular implies [2, Theorem 4.3] that, under suitable integrability conditions,
\[
\mathbb{E} \left[ e^{uV_T} \bigg| \mathcal{F}_t \right] = \exp \left\{ \phi(T - t) + \psi(T - t)V_t \right\},
\]
for any \( 0 \leq t \leq T \), where the two functions \( \phi \) and \( \psi \) satisfy the system of Riccati equations
\[
\dot{\psi}(t) = b_1 \psi(t) + \frac{a_1}{2} \psi^2(t) \quad \text{and} \quad \dot{\phi}(t) = b_0 \psi(t) + \frac{a_0}{2} \psi(t)^2,
\]
with boundary conditions \( \psi(0) = u, \phi(0) = 0 \). Assumption 2.1(i) is again borrowed from [2], and refer the interested reader to this paper for examples of kernels satisfying this condition. Most examples so far in quantitative finance, such as power-law kernel \( K(t) \equiv t^{H-1/2} \) and Gamma kernel \( K(t) \equiv t^{H-1/2} e^{-Mt} \), for \( H \in (0, 1), \lambda > 0 \), fit into this framework. Finally, Assumption 2.1(vi) allows the representation
\[
S_t = S_0 \exp \left\{ \int_0^t \left( \mu_u - \frac{I^2(u, S_u, V_u)}{2} \right) du + \int_0^t l(u, S_u, V_u) dW_u \right\}
\]
to be valid since the stochastic integrand is square integrable, by virtue of
\[
\int_0^t \mathbb{E} \left[ I^2(u, S_u, V_u) \right] du \leq \int_0^t \mathbb{E} \left[ \mathcal{L}^2(u, S_u) \varsigma^2(V_u) \right] du \leq C^2_\mathcal{L} C^2_\varsigma \int_0^t \mathbb{E} \left[ (1 + |V_u|^{2p_\varsigma}) \right] du.
\]
Remark 2.2. The assumption on the leverage function $\mathcal{L}$ may look restrictive at first, especially since, when inferring it from market data through (2), it may not be bounded; but it is customary to truncate it for large values of the underlying, and we follow this convention throughout.

Remark 2.3. One could consider a slightly different setup as (21), where the kernel does not apply to the whole dynamics of the process $V$, but only to the diffusion part, in the spirit of diffusions driven by Volterra Gaussian noises [8, 31, 3]. In the simple case, following the seminar paper by Comte and Renault [37], the variance process is the unique strong solution to the Volterra stochastic equation

$$(5) \quad V_t = V_0 e^{bt} + a \int_0^t e^{b(t-u)} dB^H_u,$$

where $B^H$ is a fractional Brownian motion with Hurst exponent $H \in (0,1)$, admitting the representation $B^H_t = \int_0^t K(u-t) dB_u$ for some standard Brownian motion $B$ with the same filtration [10] as $B^H$. The process $V$ in (3) is a continuous Gaussian process with finite variance, and Fernique’s estimate [28] yields that $E \left[ \exp \left( \alpha \sup_{t \in [0,T]} |V_t|^p \right) \right]$ is finite for any $\alpha, T > 0$ and $p < 2$, so that for any $\alpha > 0$,

$$E \left[ \exp \left( \alpha \int_0^t \ell^2(u, S_u, V_u) \, du \right) \right] \leq E \left[ \exp \left( \alpha C^2_\xi C^2_\beta \int_0^t (1 + |V_u|^{2p}) \, du \right) \right] \leq E \left[ \exp \left( \alpha C^2_\xi C^2_\beta \left( 1 + \sup_{u \in [0,t]} |V_u|^{2p} \right) \right) \right]$$

is finite and (11) holds.

2.2. Market completeness and arbitrage freeness. In the general case of non-zero correlation, the presence of the two noises renders the market incomplete. In order to be able to use the system (ii) for pricing purposes, we need to show how to complete the market, and to check for the existence of some probability measure under which the stock price is a true martingale. The latter issue was solved for the rough Heston model by El Euch and Rosenbaum [28], while the rough Bergomi case with non-positive correlation was recently proved by Gassiat [31]. We show that this still holds in our framework.

Theorem 2.4. The market is incomplete and free of arbitrage.

Proof. We wrote (iii) under the historical measure $\mathbb{P}$, but for pricing purposes, we need it under the pricing measure $\mathbb{Q}$. Introduce the market prices of risk as adapted processes $(\lambda_t)_{t \geq 0}$ and $(\beta_t)_{t \geq 0}$ such that

$$(6) \quad \rho \lambda_t + \beta_t = \frac{\mu_t - r_t}{l(t, S_t, V_t)},$$

which is well defined since the denominator is strictly positive by Assumption 2.4(ii)-(v). We now define the probability measure $\mathbb{Q}$ via its Radon-Nikodym derivative

$$\frac{d\mathbb{Q}}{d\mathbb{P}} \bigg|_{\mathcal{F}_t} = \exp \left\{ -\frac{1}{2} \int_0^t \left( \lambda^2_s + \beta^2_s \right) ds - \int_0^t \left( \lambda_s dB_s + \beta_s dB^b_s \right) \right\},$$

so that Girsanov’s Theorem [34, Chapter 3.5] implies that the processes $B^Q$ and $B^{Q, \perp}$ defined by

$$B_t^Q := B_t + \int_0^t \lambda_u du \quad \text{and} \quad B_t^{Q, \perp} := B_t^b + \int_0^t \beta_u du$$

are orthogonal Brownian motions under $\mathbb{Q}$. Therefore, under $\mathbb{Q}$, the stock price satisfies

$$(7) \quad S_t = S_0 + \int_0^t r_u S_u du + \int_0^t l(u, S_u, V_u) \left( \rho \, dB^Q_u + \beta du \right),$$

$$V_t = V_0 + \int_0^t K(t-u) \left( \hat{b}(V_u) du + \xi(V_u) dB^Q_u \right),$$

where, under $\mathbb{Q}$, the drift of the variance process is now of the form $\hat{b}(V_t) = b(V_t) - \lambda_t \xi(V_t)$. Introducing the Brownian motion $W^Q$ (under $\mathbb{Q}$), the first SDE then reads

$$S_t = S_0 + \int_0^t r_u S_u du + \int_0^t l(u, S_u, V_u) dW^Q_t.$$
The proposition then follows directly from Novikov’s criterion thanks to the assumptions on the function \( l(\cdot) \) in Assumption 2.1(iii)-(vi).

\[ \square \]

**Remark 2.5.** Following [22], we could in fact partially relax the assumption on the function \( l(\cdot) \). Assume for example that \( l(t,S,v) = \Sigma(t,S)\varsigma(v) \), but with Assumption 2.1 replaced by \( \varsigma(v) = \exp(\eta v) \), for some \( \eta > 0 \). Then, recalling that \( \rho \leq 0 \), the proof of [22, Theorem 1], using a localisation argument with the increasing sequence of stopping times \( \tau_n := \inf\{ t > 0 : V_t = n \} \), remains the same, and the stock price is a true martingale. This in particular means that a local volatility version of the rough Bergomi model [3, 22] falls within our current setup.

### 3. Pricing via Deep PPDEs

We shall from now on only work under the risk-neutral measure and, with a slight abuse of notations, write \( b \) instead of \( \hat{b} \) for the drift of the variance process (equivalently taking the market price of volatility risk to be null in Theorem 2.1). In the classical Itô diffusion setting, where the kernel \( K \) is constant, Feynman-Kac formula transforms the pricing problem from a probabilistic setting to a PDE formulation. The formulation of our rough local stochastic volatility model [22] goes beyond this scope since the system is not Markovian any longer. We adapt here the results developed by Viens and Zhang [22] to show that the pricing problem is equivalent to solving a path-dependent PDE.

#### 3.1. Pricing via path-dependent PDEs

We first start with the following lemma, which hints shows, not surprisingly, that the option price should be viewed, not as a function of the state variable at a fixed given time, but as a functionals over paths. The argument is not new, and versions have already appeared for the rough Heston model in [22] and in the context of Itô’s formula for stochastic Volterra equations in [22]. Consider a European option with maturity \( T \) and payoff function \( g(\cdot) \) written on the stock price \( S \). By standard no-arbitrage arguments, from Theorem 2.1, its price at time \( t \in [0,T] \) reads

\[ (8) \quad P_t = \mathbb{E}[g(S_T)|\mathcal{F}_t]. \]

**Lemma 3.1.** There exists a function \( P : [0,T] \times [0,\infty) \times C([0,T] \to \mathbb{R}) \), such that, for any \( t \in [0,T] \),

\[ P_t = P(t,S_t,\Theta^t_{\leq t} \mid \mathcal{F}_t), \]

where \( \Theta^t \in \mathcal{F}_t \) and is given, for any \( u \in [t,T] \), by

\[ (9) \quad \Theta^t_u := \mathbb{E}\left[ V_u - \int_t^u K(u-r)b(r,V_r)dr \bigg| \mathcal{F}_t \right]. \]

**Proof.** For any \( t \in [0,T] \) and \( u \in [t,T] \), we can write, from (22)

\[
V_u = V_0 + \int_0^u K(u-r) (b(V_r)dr + \xi(V_r)dB_r) \\
= V_0 + \int_t^u K(u-r) (b(V_r)dr + \xi(V_r)dB_r) + \int_t^u K(u-r) (b(V_r)dr + \xi(V_r)dB_r),
\]

so that \( \Theta^t \) is clearly \( \mathcal{F}_t \)-measurable and satisfies (22). We can therefore write the option price as

\[
\mathbb{E}[g(S_T)|\mathcal{F}_t] = \mathbb{E}\left[ g\left( S_t + \int_t^T r_u S_u du + \int_t^T l\left(u, S_u, \Theta^t_u + I^t_u\right) dW_u \right) \bigg| \mathcal{F}_t \right],
\]

and the lemma follows since \( I^t \notin \mathcal{F}_t \).

\[ \square \]

For any \( t \in [0,T] \), the curve \( \Theta^t \) (or equivalently the process infinite-dimensional process \( (\Theta^t)_{t \in [0,T]} \)) plays a fundamental role. It does not exactly represent the forward variance curve, which is used in [22], and we refer to [22] for the precise correspondence between the two. Note that \( \Theta^t_u = v_u \) for \( u \in [0,t] \), in particular \( \Theta^t_t = v_t \). The following theorem is the main result here (proved in Appendix 2), and shows how to extend the classical Feynman-Kac formula to the path-dependent case. From now on, we shall adopt the notation

\[ (10) \quad K^t := K(-t), \quad \text{for any } t \geq 0, \]

to denote the curve \( K^t \) seen at time \( t \). It will become clear in the following theorem how this becomes handy.
The option price (8) is the unique solution to the linear path-dependent PDE
\begin{equation}
\left( \partial_t + \mathcal{L}_x + \mathcal{L}_{xx} + \mathcal{L}_{\omega} + \mathcal{L}_{\omega\omega} - r_t \right) P(t, S_t, \Theta^t) = 0,
\end{equation}
for \( t \in [0,T] \), with boundary condition \( P(T, S_T, \Theta^T) = g(S_T) \), where the differential operators are defined as
\[
\mathcal{L}_{xx} := \rho(t, x, v) \xi(t) x \langle \partial_x, K^t \rangle, \quad \mathcal{L}_{\omega} := \frac{1}{2} l(t, x, \Theta^t)^2 x^2 \partial_x^2, \quad \mathcal{L}_x := r_t x \partial_x,
\]
for some fixed integer \( n \).

The pricing PDE (11) has both state-dependent terms, involving derivatives with respect to \( x \), and path-dependent ones, involving functional derivatives with respect to \( \omega \). In order to streamline the presentation, we defer to Appendix A the precise framework, borrowed from [22], to define these derivatives. Without essential loss of understanding for the rest of the analysis, the reader can view them basically as Gâteaux derivatives (with some regularisation due to the singularity of the kernel along the diagonal). Path-dependent PDEs have been studied in great details, and we refer the interested reader to [22, 24, 26]. However, numerical schemes for such equations are rather scarce, and the only approaches we are aware of is the extension of Barles and Souganidis’ monotone scheme [12] to the path-dependent case by Zhang and Zhou [23], the convergence of which was proved by Ren and Tan [30]. However, the actual implementation of this scheme in the PPDE context is far from trivial, and we consider a different route here, more amenable to computations in our opinion. We first discretise the PPDE along some basis of functions, reducing the infinite-dimensional problem to a finite-, yet high-, dimensional problem. High-dimensional PDEs suffer from the so-called curse of dimensionality, and are notoriously difficult to solve. We then adopt the deep learning approach recently developed by Hen and Jentzen [11] to solve this system of PDEs.

3.2. Discretisation of the PPDE. For each \( t \in [0,T] \), we consider a basis \( (\psi^t_i)_{i=1,\ldots,n} \) of càdlàg functions, for some fixed integer \( n \), and use it to approximate \( \Theta^t \) and \( K^t \) by
\[
\hat{\Theta}^t := \sum_{i=1}^n \theta^t_i \psi^t_i \quad \text{and} \quad \hat{K}^t := \sum_{i=1}^n \kappa^t_i \psi^t_i,
\]
for some sequence of real coefficients \( \theta^t := (\theta^t_i)_{i=1}^n \) and \( \kappa^t := (\kappa^t_i)_{i=1}^n \). Since \( K^t \in \mathcal{D}_t \), the space of càdlàg functions on \([t,T]\) (see Appendix A.2), then, from Definition A.3,
\[
\langle \partial_x P \left( t, x, \Theta^t \right), K^t \rangle := \partial_x P \left( t, x, \Theta^t + \varepsilon K^t \right) \bigg|_{\varepsilon=0} = \partial_x P \left( t, x, \Theta^t \right) \bigg|_{\varepsilon=0},
\]
and we can introduce the following approximations of the path derivatives along the direction \( K^t \):
\[
\langle \partial_x P \left( t, x, \hat{\Theta}^t \right), \hat{K}^t \rangle := \partial_x P \left( t, x, \hat{\Theta}^t + \varepsilon \hat{K}^t \right) \bigg|_{\varepsilon=0} = \partial_x P \left( t, x, \sum_{i=1}^n \left( \theta^t_i + \varepsilon \kappa^t_i \right) \psi^t_i \right) \bigg|_{\varepsilon=0} = \partial_x \hat{P} \left( t, x, (\theta^t_i + \varepsilon \kappa^t_i) \right) = \partial_x \hat{P} \left( t, x, \theta^t \right) \kappa^t,
\]
where the function \( \hat{P} \) now acts on \([0,T] \times [0,\infty) \times \mathbb{R}^n \). Likewise, for the second functional derivative, we have the approximation
\[
\langle \partial^2_x \left( t, x, \hat{\Theta}^t \right), \hat{K}^t \rangle := \sum_{i,j=1}^n \partial^2_{\theta^t_i \theta^t_j} \hat{P} \left( t, x, \theta^t \right) \kappa^t_i \kappa^t_j = (\kappa^t)^\top \Delta \theta^t \hat{P} \left( t, x, \theta^t \right) \kappa^t,
\]
and finally the cross derivatives can be approximated similarly as
\[
\langle \partial_{x,\omega} \left( t, x, \hat{\Theta}^t \right), \hat{K}^t \rangle := \partial_x \nabla_{\theta^t} \hat{P} \left( t, x, \theta^t \right) \cdot \kappa^t.
\]
The PPDE (12) therefore becomes
\[
\left( \partial_t + \mathcal{L}_x + \mathcal{L}_{xx} + \sum_{i=1}^n \mathcal{L}_{x \theta^t_i} + \sum_{i=1}^n \mathcal{L}_{\theta^t_i} + \sum_{i,j=1}^n \mathcal{L}_{\theta^t_i \theta^t_j} - r_t \right) \hat{P} = 0,
\]
where the differential operators are defined, for each \( i, j = 1, \ldots, n \), as
\[
\mathcal{L}_{x_i} := \rho \left( t, x, \theta^i \right) \xi(\theta^i) \kappa^i \partial_{x_i}, \quad \mathcal{L}_{xx} := \frac{1}{2} \left( t, x, \theta^i \right)^2 \partial_x^2, \quad \mathcal{L}_x := r_t x \partial_x,
\]
\[
\mathcal{L}_{\theta^i \theta^j} := \frac{\xi(\theta^i)^2}{2} \kappa^i \kappa^j \partial_{\theta^i \theta^j}, \quad \mathcal{L}_{\theta^i} := b(\theta^i) \kappa^i \partial_{\theta^i}.
\]

We can rewrite this system in a more concise way as
\[
\partial_t \hat{P} + \frac{1}{2} \text{Tr} \left( \Sigma \cdot \Sigma^T \cdot \Delta \hat{P} \right) + \mu \cdot \nabla \hat{P} - r_t \hat{P} = 0,
\]
where \( \mu(t, x, \theta^i) := (x t, b(\theta^i) \kappa_1, \ldots, b(\theta^i) \kappa_n)^T \) and
\[
\Sigma(t, x, \theta^i) \cdot \Sigma(t, x, \theta^i)^T := \begin{pmatrix}
\frac{1}{2} \rho(t, x, \theta^i)^2 & \rho(t, x, \theta^i) \xi(\theta^i) \kappa^i_1 & \cdots & \rho(t, x, \theta^i) \xi(\theta^i) \kappa^i_n \\
\rho(t, x, \theta^i) \xi(\theta^i) \kappa^i_1 & \xi(\theta^i)^2(\kappa^i_1)^2 & \cdots & \xi(\theta^i)^2 \kappa^i_1 \kappa^i_n \\
\vdots & \vdots & \ddots & \vdots \\
\rho(t, x, \theta^i) \xi(\theta^i) \kappa^i_n & \xi(\theta^i)^2 \kappa^i_n & \cdots & \xi(\theta^i)^2(\kappa^i_n)^2
\end{pmatrix}.
\]

**Remark 3.3.** The simplest example is to consider piecewise constant curves \( \psi^i_t = 1_{\delta^i_t} \), for \( i = 1, \ldots, n \), where \( (\delta^i_t)_{t=1}^n \) represents a mesh of the interval \([t, T]\), say of the form \( \delta^i_t = [t_{i-1}, t_i] \). In that case, we can consider \( \theta^i_t = \theta^i_{t_i} \) and \( \kappa^i_t = K(t_i - t) \).

There is an interesting connection between the functional Itô formula in Theorem A.14 and backward stochastic differential equations. Following [23], consider the multidimensional BSDE
\[
\begin{cases}
X_t = \xi + \int_0^t \pi_t(r, X_r)dr + \int_0^t \Sigma(r, X_r)dW_r, \\
Y_t = g(X_T) + \int_T^t f(r, X_r, Y_r, Z_r)dr - \int_T^T Z_r^T dW_r,
\end{cases}
\]
on some filtered probability space \( \Omega, \mathcal{F}, (\mathcal{F}_t)_{t\geq 0}, \mathbb{P} \) supporting a \( d \)-dimensional Brownian motion \( W \). The solution process \((X, Y, Z)\) takes values, at each point in time, in \( \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^d \). Consider further the PDE
\[
\partial_t u(t, x) + \frac{1}{2} \text{Tr} \left( \Sigma(t, x) \Sigma(t, x)^T \Delta u(t, x) \right) + \pi(t, x) \nabla u(t, x) + f(t, x, u(t, x), \Sigma(t, x)^T \nabla u(t, x)) = 0,
\]
with terminal boundary condition \( u(T, x) = g(x) \). In the regular (Assumption A.1(i)) case, as shown in [23], assuming that (13) has a classical solution \( u(\cdot) \) in \( C^2_B(\Lambda) \) (see Appendix A.2), then the couple \((Y, Z)\) defined as \( Y_t := u(t, X_t) \) and \( Z_t := \Sigma(t, X_t)^T \cdot \nabla u(t, X_t) \) is the solution to (14). In light of this result, and following up on the main idea in [11], if the solution to the option problem satisfies (13), it also solves the BSDE (14), namely
\[
\hat{P}(t, S_t, \theta^T) = \hat{P}(T, S_T, \theta^T) + \int_t^T r_u \hat{P}(u, S_u, \theta^u) du - \int_t^T \Sigma(u, S_u, \theta^u)^T \cdot \nabla \hat{P}(u, S_u, \theta^u) dW_u,
\]
or, written in forward form,
\[
\hat{P}(t, S_t, \theta^T) = \hat{P}(0, S_0, \theta^0) - \int_0^t r_u \hat{P}(u, S_u, \theta^u) du + \int_0^t \Sigma(u, S_u, \theta^u)^T \cdot \nabla \hat{P}(u, S_u, \theta^u) dW_u.
\]

### 3.3. Neural network structure

We now introduce the neural network structure that will help us solve the high-dimensional pricing problem above. We concentrate on the setting in Remark 3.3.

#### 3.3.1. Simulation of the network inputs

Recall from (11) that the volatility process is given by
\[
V_t = V_0 + \int_0^t \kappa(t - r) \left( b(V_r)dr + \xi(V_r)dB_r \right),
\]
and for any \( t \in [0, T] \), the \( \mathcal{F}_t \)-measurable random variable \( \Theta^T \) introduced in (11) is given by
\[
\Theta^T_u = V_0 + \int_0^t \kappa(u - r) \left( b(V_r)dr + \xi(V_r)dB_r \right),
\]
for \( u \geq t \).
We discretise both time directions (pricing time and maturity) along the same square grid \( (t_i = \frac{it}{n})_{0 \leq i \leq n} \), with \( \Delta_i := t_i - t_{i-1} \). The matrix \( \left( \theta_j^i \right)_{0 \leq j \leq n} \) is given by

\[
\theta_j^i = \begin{cases} 
V_j, & \text{if } 0 \leq j \leq i, \\
\theta_{j-1}^i + \int_{t_{j-1}}^{t_j} K(t_j - u) \left( b(V_u) du + \xi(V_u) dB_u \right), & \text{if } j > i.
\end{cases}
\]

Discretising the stochastic integral here is not trivial, and we appeal to the hybrid scheme developed in [10].

Over the interval \([t_i, t_{i-1}]\), the stochastic integral can be decomposed as

\[
\int_{t_{i-1}}^{t_i} K(t_j - u) \xi(V_u) dB_u = \int_{t_{i-1}}^{t_i} K(t_j - u) \xi(V_u) dB_u - \int_{t_i}^{t_j} K(t_j - u) \xi(V_u) dB_u.
\]

Now, the second term on the right-hand side reads (we use the hybrid scheme truncated at the first level)

\[
\int_{t_i}^{t_j} K(t_j - u) \xi(V_u) dB_u = \sum_{k=1}^{j-1} \xi(V_{t_{j-k}}) \int_{t_{j-k}}^{t_{j-1}} K(t_j - u) dB_u
\]

\[
= \xi(V_{t_{j-1}}) \int_{t_{j-1}}^{t_j} K(t_j - u) dB_u + \sum_{k=2}^{j-1} \xi(V_{t_{j-k}}) \int_{t_{j-k}}^{t_{j-1}} dB_u,
\]

where the term \( \xi(V_u) \) is approximated by its left point (it is obvious that \( \overline{B}_j \) does not depend on \( k \), so we omit it the latter from the notation), with the coefficients \( (b_k) \) chosen appropriately (see also Remark 3.4 below). The couple \( \left( \overline{B}_j, \overline{B}_j \right) \) forms a two-dimensional Gaussian vector, with covariance matrix \( \Sigma \) given by

\[
\Sigma_{ij} = \int_{t_{i-1}}^{t_i} K(t_j - u)^2 du,
\]

\[
\Sigma_{12} = \int_{t_{i-1}}^{t_i} K(t_j - u) dB_u,
\]

\[
\Sigma_{21} = \int_{t_{i-1}}^{t_i} K(t_j - u) dB_u,
\]

\[
\Sigma_{22} = \frac{1}{n},
\]

\[
\Sigma_{11} = \int_{t_{i-1}}^{t_i} K(t_j - u)^2 du,
\]

\[
\Sigma_{12} = \frac{1}{n}.
\]

Remark 3.4. In the power law case \( K(t) = t^{H-\frac{1}{2}} \) with \( H \in (0,1) \), then

\[
\Sigma = \begin{pmatrix}
\frac{1}{2}Hn^{2H} & 1 \\
\frac{1}{(H+\frac{1}{2})n^{H+\frac{1}{2}}} & \frac{1}{n-1}
\end{pmatrix},
\]

and, as shown in [10], the coefficients \( (b_k) \) are explicitly computed as

\[
b_k := \left( k^{H+\frac{1}{2}} - (k-1)^{H+\frac{1}{2}} \right) \frac{1}{H+\frac{1}{2}}.
\]

For the stock price, starting from \( S_{t_0} = S_0 \), we use the discretised explicit form, for \( i = 1, \ldots, n \),

\[
S_{t_i} = S_{t_{i-1}} \exp \left\{ \left( r_{t_{i-1}} - \frac{1}{2} l(t_{i-1}, S_{t_{i-1}}, V_{t_{i-1}})^2 \right) \Delta_i + l(t_{i-1}, S_{t_{i-1}}, V_{t_{i-1}}) \left( \rho \Delta B_{t_i} + p \Delta B_{t_i} \right) \right\},
\]

3.3.2. Euler discretisation scheme. We are iteratively computing the price of the option at each time step. Both \( \hat{P}_0 \) and \( (\nabla \hat{P}_0) \) are the initial price and gradient that will be optimised with the weights of the network. On the two-dimensional grid \( (t_i, t_j)_{0 \leq i, j \leq n} \), we can discretise the forward stochastic equation (17) as

\[
\begin{align*}
\hat{P}(t_0, S_{t_0}, \theta^0) &= \hat{P}_0, \\
\nabla \hat{P}(t_0, S_{t_0}, \theta^0) &= \left( \nabla \hat{P} \right)^0, \\
\hat{P}(t_{i+1}, S_{t_{i+1}}, \theta^{i+1}) &= (1 + r_i \Delta_i) \hat{P}(t_i, S_{t_i}, \theta^i) + \Sigma \left( t_i, S_{t_i}, \theta^i \right)^\top \nabla \hat{P}(t_i, S_{t_i}, \theta^i) \Delta W_{t_i}, \\
\hat{P}(t_n, S_{t_n}, \theta^n) &= g(S_T),
\end{align*}
\]
where \( W_t = \left( B_{t_1}^Q, B_{t_1}, \ldots, B_{t_i} \right) ^\top \in \mathbb{R}^n \) and the diffusion matrix reads, for any \( 1 \leq i \leq n, \)

\[
\Sigma(t, S_t, \theta^t) = \begin{pmatrix}
\rho l(t, S_t, V_t) & \rho l(t, S_t, V_t) & \cdots & 0 \\
0 & \xi(V_t)K(t_i - t_0) & \cdots & \\
& \ddots & \ddots & 0 \\
0 & 0 & \cdots & 0 \xi(V_t)K(t_i - t_n)1_{i>n}
\end{pmatrix} \in \mathcal{M}_{n+2,n+2}(\mathbb{R}).
\]

In particular, for any \( j = 2, \ldots, n + 2, \) the \( j \)-th diagonal term reads

\[
\Sigma(t, S_t, \theta^t)_{j,j} = \xi(V_t)K(t_i - t_{j-2})1_{i>j-2}.
\]

Discretising the backward SDE would give

\[
\begin{cases}
\hat{P}(t_n, S_{t_n}, \theta^{t_n}) = g(S_T), \\
\hat{P}(t_i, S_{t_i}, \theta^{t_i}) = (1 - r_{t_{i+1}} \Delta_{1+1}) \hat{P}(t_{i+1}, S_{t_{i+1}}, \theta^{t_{i+1}}) - \Sigma(t, S_t, \theta^t)^\top \nabla \hat{P}(t_i, S_{t_i}, \theta^t) \Delta W_t, \\
\hat{P}(t_n, S_{t_n}, \theta^{t_n}) = g(S_T),
\end{cases}
\]

We shall follow here this backward approach as it is more natural for pricing exotic (path-dependent) derivatives, such as Bermudan or American options.

3.4. Sub-network. At each time step, the price as well as the model parameters are known. The only unknown is the term \( \Sigma(t, S_t, \theta^t)^\top \nabla \hat{P}(t_i, S_{t_i}, \theta^t) \) involving the gradient and the diffusion matrix. We therefore use a neural network to infer its value:

\[
\begin{array}{cccc}
\hat{P}(t_{i-1}, S_{t_{i-1}}, \theta^{t_{i-1}}) & \rightarrow & \hat{P}(t_i, S_{t_i}, \theta^t) & \rightarrow & \hat{P}(t_{i+1}, S_{t_{i+1}}, \theta^{t_{i+1}}) & \rightarrow & \cdots & \hat{P}(t_n, S_{t_n}, \theta^{t_n}) \\
\nabla \hat{P}(t_{i-1}, S_{t_{i-1}}, \theta^{t_{i-1}}) & \rightarrow & \nabla \hat{P}(t_i, S_{t_i}, \theta^t) & \rightarrow & \nabla \hat{P}(t_{i+1}, S_{t_{i+1}}, \theta^{t_{i+1}}) & \rightarrow & \cdots & \nabla \hat{P}(t_n, S_{t_n}, \theta^{t_n})
\end{array}
\]

The input is composed of one input layer with the value of the processes at time \( t_i \). Each hidden layer is computed by multiplying the previous layer by the matrix of weights \( \mathbf{w} \) and adding a bias \( \delta \). After computing
a layer, we apply a batch normalisation by computing the mean \(m\) and the standard deviation \(s\) of the layer, and by applying the following linear transformation to each element of the layer:
\[
T(x) := \gamma \frac{x - m}{s} + \beta,
\]
where the scale \(\gamma\) and the offset \(\beta\) are to be calibrated. We also use the ReLu activation function \(a(x) = x_+\) on each element of the layer.

3.5. **Optimisation of the algorithm.** In the algorithm, \(n_l\) denotes the number of layers per sub-network, \(n_N\) the number of neurons per layer, \(n_B\) the number of batches used to separate the samples, \(N\) the size of each batch and \(\text{epoch}\) shall denote the number of times all the samples are fed to the training algorithm. We finally introduce the following loss function that we aim to minimise:

\[
L \left( w, \delta, \beta, \gamma, \hat{P}_0, (\nabla \hat{P})_0 \right) := \mathbb{E} \left[ \left( g \left( S_{t_n} \right) - \hat{P} \left( (t_i)_{1 \leq i \leq n}, (S_{t_i})_{1 \leq i \leq n}, (\theta^i)_{1 \leq i \leq n} \right) \right)^2 \right],
\]

or, in fact, its version on the sample,

\[
\hat{L} \left( w, \delta, \beta, \gamma, \hat{P}_0, (\nabla \hat{P})_0 \right) := \frac{1}{N} \sum_{k=1}^{N} \left[ g \left( S^k_{t_n} \right) - \hat{P} \left( (t_i)_{1 \leq i \leq n}, (S^k_{t_i})_{1 \leq i \leq n}, (\theta^i)_{1 \leq i \leq n} \right) \right]^2.
\]

It represents the variance of the initial price found by backward iterations for each simulated path. Since the initial price is unique and deterministic, minimising its variance is natural good way to compute the initial price. Regarding the optimisation itself, we use the Adaptive Moment Estimation method \([3]\) for the first iterations, and switch to the stochastic gradient method, with slower but more stable convergence properties. The different parameters to calibrate are then
- The weights \(w\) and biases \(\delta\) for each layer of each sub-network;
- The \(\beta\) and \(\gamma\) in the batch normalisation;
- The initial price \(\hat{P}_0\);
- The initial gradient \((\nabla \hat{P})_0\).

4. **Numerical results**

4.1. **The Black-Scholes case.** We first test the accuracy of the method on a European Call option with strike \(K = 100\) and maturity \(T = 1\) in the Black-Scholes model \(dS_t = S_t(\mu dt + \sigma dW_t)\) with \(S_0 = 100\), volatility \(\sigma = 5\%\), and interest rate \(r = 5\%\). To train the model, we use different hyper-parameter configurations and study the impact of each parameter on the loss function in \([14]\). The default configuration is the following:
- 1 epoch;
- 1000 iterations;
- Batch size = 5000;
- 2 hidden layers;
- 10 neurons per layer;
- Activation function for the hidden layers: ReLu;
- Activation function for the hidden layers: None.

4.1.1. **Impact of the hyper-parameters on the loss function.** We first study the influence of each parameter of the model on the loss function \([14]\), and summarises the results in Figure \(\text{II}\). As a function of the batch size, the loss function (on a logarithmic scale) flattens out rapidly, so that one needs to increase the batch size considerably to improve the accuracy, which is however limited by computing power. One solution is to use mini-batches that accumulate the gradients, and apply them altogether for a chosen number of mini-batches. This allows to train the model with a very large batch. However this is hugely time consuming since the network has to process the mini-batches one by one. For 1000 iterations, it is interesting to note that the loss function reaches a minimum for two hidden layers. Fewer layers would prevent the network to approximate the gradient of the price accurately, while too many would give to much freedom to the model. As a function of the number of neurons per layer, the loss function decays exponentially fast, and six neurons per layer seems enough for good accuracy. When plotting the loss in terms of the number of time steps, the convexity of the graph suggests that sixty time steps are sufficient. However, since time steps also determine the
number of neural networks to infer each gradient (and hence the number of weights to calibrate), we have to limit their number.

![Image](https://ssrn.com/abstract=3400035)

(a) Loss function vs batch size (on a logarithm scale).

(b) Loss function vs number of hidden layers.

(c) Loss vs number of neurons per layer.

(d) Loss function vs number of time steps.

**Figure 1.** Influence of the hyper-parameters on the loss function.

### 4.1.2. Impact of the hyper-parameters on the price accuracy.

To assess the accuracy of the computed price itself, we compute the mean and standard deviation of the price obtained in the last 100 iterations, the goal being to determine the configuration with the smallest standard deviation. Following the previous section, we consider 50 time steps, two hidden layers with six neurons per layer, and since the batch sizes we are taking are high, we limit ourselves to 200 iterations. Since we integrated gradient accumulation to our solver, we can now take as input very large batches without memory issues. Table 2 shows the influence of the batch size on the price accuracy. Even though the standard deviation decreases while increasing the batch size, it does not quite converge, and the batch size does not really affect the accuracy of the convergence. This can be explained by the relatively low number of iterations, and by the samples that may not be representative enough (high discrepancy) to yield accurate convergence. An interesting observation is that the number of steps (Table 2) seems to have relatively little influence on the convergence of the accuracy. The latter could in principle be improved in many ways, and numerical tests seem to disqualify the following approaches:

- using layer normalisation instead of batch normalisation;
- using a dropout rate to drop randomly a percentage of samples at each layer;
- using different activation functions;
- using weight regularisation by adding an $L^2$-norm penalty for the weights.

On the positive side, though, we used quasi-random (Sobol) sequences instead of pseudo-random generators, and found that the algorithm converges faster. Note that, for speed purposes, the Sobol sequences can be pre-computed and stored, and then loaded when needed by the network. We so far always considered the same size for the validation sample and for the training batch. In the backward method (our main choice as it allows us to price callable and Bermudan options), the price is computed by minimising its variance through a Monte-Carlo-like method. Hence, increasing the size of the validation sample naturally decreases the variance. We then choose a very large validation sample ($10^6$) and a relatively smaller training batch size (of order $10^4$).
### 4.1.3. Pricing Call options.

We compute the prices of a European Call option with the previous parameters for different strikes, and compare these in Table 3 to the true Black-Scholes values. The errors seem to vary depending on whether the option is in or out of the money. For an option with larger maturity, but with the same precision, one needs to keep the number of time steps per year constant and hence increase the number of time steps, which consequently increases the computation time.

| Strike | 80  | 85  | 90  | 95  | 100 | 105 | 110 | 115 | 120 |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| DL Price | 23.90 | 19.14 | 14.39 | 9.67 | 5.28 | 2.05 | 0.51 | 0.07 | 0.01 |
| True price | 23.90 | 19.14 | 14.39 | 9.67 | 5.28 | 2.05 | 0.51 | 0.07 | 0.01 |
| Absolute error (10^{-4}) | 15 | 12 | 9.6 | 3.8 | 3 | 3.8 | 1.1 | 0.85 | 0.18 |
| Relative error (10^{-4}) | 0.62 | 0.63 | 0.67 | 0.4 | 0.56 | 1.8 | 1.7 | 11 | 27 |

Table 3: Deep Learning vs true prices (via the Black-Scholes formula) and errors.

### 4.2. Rough stochastic local volatility models.

We now investigate the accuracy of our algorithm in the more interesting case of rough local stochastic volatility models. As mentioned in the introduction, local stochastic volatility models have become the standard on Equity markets, with each bank using its own version. We consider here the model (III) with

\[ b(V) = \kappa(V_\infty - V), \quad \xi(V) = \nu V, \quad \varsigma(V) = e^{V}, \quad K(t) = \frac{t^{H-\frac{1}{2}}}{\Gamma(H + \frac{1}{2})}, \]

with \( \kappa, V_\infty, \nu > 0, H \in (0, 1) \), and no interest rates, so that the variance process \( (V_t)_{t \geq 0} \) is a fractional Ornstein-Uhlenbeck process. We acknowledge that the function \( \varsigma(\cdot) \) does not quite satisfy Assumption 2.1(v), but leave this for future theoretical considerations. Note in passing that the true martingale property of the stock price in the rough Bergomi model—at least for non positive correlation—was recently proved by Gassiat [31], and similar arguments could be used to extend the result to the current setting. The OU process can become negative, but, because of the function \( \varsigma(\cdot) \), the actual instantaneous volatility of the stock price is guaranteed to remain strictly positive at all times. The local volatility component is more delicate, and we consider the usual approach introduced in Section 2.1. The Dupire [18] local volatility function \( \sigma_1 \) can be recast in terms of the implied total variance \( w(k,T) := \sigma_{implied}^2(k,T)T \) through the identity [32, Chapter 1]

\[ \sigma^2_{implied}(k,T) = \frac{\partial_T w(k,T)}{\varrho(k,T)}, \]

where \( k \) represents the log-moneyness, and

\[ \varrho(k,T) := \left( 1 - \frac{k \varrho_k w}{2w} \right)^2 - \frac{(\varrho_k w)^2}{4} \left( \frac{1}{4} + \frac{1}{w} \right) + \frac{\varrho_{kk} w}{2} \right|_{(k,T)}. \]
This relationship is not trivial to apply in practice though, as the observed implied volatility surface is not smooth, and only observed discretely. We assume here that the surface \( w(\cdot, \cdot) \) is given through the SSVI configuration developed by Gatheral and Jacquier \[33\], namely

\[
w(k, T) = \frac{\theta_T}{2} \left\{ 1 + \varphi(\theta_T)k + \sqrt{(\varphi(\theta_T)k + \varphi)^2 + 1 - \varphi^2} \right\},
\]

where \( \varphi \in [-1, 1], (\theta_t)_{t \geq 0} \) represents the term structure of the at-the-money total implied variance, and the function \( \varphi(\cdot) \) provides a backbone. In particular the so-called Heston-like parameterisation \[33, Example 4.1\]

\[
\varphi(\vartheta) = \frac{1}{\lambda \vartheta} \left( 1 - \frac{1 - e^{-\lambda \vartheta}}{\lambda \vartheta} \right),
\]

for some strictly positive constant \( \lambda \), was shown \[33\] to be free of static arbitrage, and therefore suitable as a starting point for market generation. Regarding the neural network, we use the default configuration

- Batch size: \( 10^4 \);
- Validation batch size: \( 2 \times 10^6 \);
- 200 iterations (epochs);
- Two hidden layers;
- 5 neurons per hidden layer;
- Between 100 and 2, 600 time steps per year depending on the maturity;
- Between 20 and 520 BSDE time steps per year depending on the maturity.

We will take the following values per default for the parameters of the model:

\[
S_0 = 100, \quad T = 0.1, \quad V_0 = \log(0.15), \quad \nu = 0.2, \quad V_\infty = 0.15, \quad \kappa = 0.5, \quad \rho = -0.4.
\]

At first, we try to recover the BS price setting \( \rho = 1, \nu = 0 \) and \( \kappa = 0 \). Note that the number of time steps controls the dimension as well as the number of sub-networks, and one needs to decrease it in order to get a convenient computation time. We then study (numerically) the influence of each parameter on the implied volatility smile, and summarise them in Figure 2. We compute the volatility smile for a given configuration and we study the impact of each model parameter. We take an extreme case with a very low correlation \( \rho = -0.95 \), and very small Hurst parameters \( H \in \{0.05, 0.01\} \). In order to compute a smile as accurate as possible we will use Call options when the strike is above the initial stock value and put options when the strike is below. This is in order to deal only with out of the money options who reflect more the time value rather than the intrinsic value. In Figure 2(a), we compute the smile for three values of correlation, negative, null and positive. The correlation has an effect on the symmetry of the smile. Indeed we observe a volatility skew when setting the volatility different than zero. The orientation of the skew (more expensive OTM Call or Put options) depends on the sign of the correlation (respectively positive and negative). Figure 2(b) looks at the influence of the volatility of volatility. When the latter is low, the smile is close to constant, as in the Black-Scholes model. An increase in the value of volatility of volatility affects the convexity of the smile as well as the level. The influence of the mean reversion can be seen in Figure 2(c), with \( H = 0.1 \). In particular, the mean reversion affects slightly the smile convexity: the higher the mean reversion, the faster the volatility reverts to its long-term mean \( V_\infty \) and the smaller the curvature. Finally, the role of the Hurst parameter can be observed in Figure 2(d), with an effect similar to that of the volatility of volatility: smaller values yields higher curvature and (slightly) higher level.

The computations require a large number of time steps for small maturities. We differentiate the process simulation time steps from the BSDE time steps. Indeed, fewer BSDE time steps are needed in order to obtain decent computation times for the BSDE solver. We compare for \( T = 0.1 \) the smiles obtained with 100 simulations steps for the processes and different numbers of BSDE steps. We observe that the number of BSDE time steps does not have a big impact for short maturities. The range of moneyness where the model smile matches the parameterised one is \([0.9; 1.15]\). Outside that range, the events are to rare for this short maturity to be able to compute a price using the BSDE method. That is why this is diverging.

4.3. Model calibration. We now provide a way to calibrate such a rough local stochastic volatility model. We assume that the mean reversion \( \kappa \), the Hurst parameter \( H \) and the initial volatility \( V_0 \) are given, while the volatility of volatility \( \nu \), the correlation \( \rho \) and the long-term variance level \( V_\infty \) need to be calibrated. The reason for this choice is practical (the dimension of the optimisation problem is reduced), but can be easily
(a) Influence of the correlation $\rho$.

(b) Influence of the volatility of volatility $\nu$.

(c) Influence of the mean reversion $\kappa$.

(d) Influence of the Hurst parameter.

FIGURE 2. Influence of the rough volatility parameters on the smile, with $T = 0.1$.

FIGURE 3. Implied volatility for different number of BSDE time steps in the rough LSV model.

justified: a good proxy for the initial variance $V_0$ is given by the short-term at-the-money smile, and the Hurst parameter can be calibrated by the maturity-decay of the at-the-money skew of the implied volatility smile; proper and rigorous explanations, via asymptotic limits and expansions, can be found in [3, 1, 24, 31, 47, 41, 52]. We perform a slice by slice calibration, via a minimisation of the difference between market smiles and of model smiles. However, with the Deep Learning method, this is extremely computer intensive (several hours), so we instead propose the following calibrating method using machine learning techniques, inspired by [17, 65]:

Electronic copy available at: https://ssrn.com/abstract=3400035
1. We generate a volatility surface on the grid \((V_\infty, \nu, \rho) \in \{0.1, 0.15, 0.2\} \times \{0.1, 0.2, 0.3\} \times \{-0.6, -0.4, -0.2\}\).

2. Training the neural network. We use the 'sub-network' of Section 3.4 as our network to train. For a given maturity, the smiles generated by the sets of training data are the input of the network, and the three parameters are the labels. Theoretically, the network needs to be fed with several thousands of samples. However, since it takes time to simulate so many smiles, we try with a small number of smiles but with a lot of epochs (number of times all the samples are fed to the network).

3. Once the weights of the neural networks are calibrated, the prediction part consists in inputting the market smile used for the calibration in the network in order to get the corresponding parameters. Note that once the samples are produced and the network is trained, the calibration part is almost instantaneous (since it consists only in multiplying the input by weights matrices).

We first train the network with only one smile associated to a set of parameters in order to configure the hyper-parameters. While testing the same smile as an input, we obtain the right parameters when using one layer and 100 epochs (the number of neurons does not seem to impact the precision). We then train the network with 26 smiles generated from the grid above. The smile not used for calibration is then used as input into the network to recover its associated parameters. Note that the system \((X_t)\) is equivalent to the fact that it can be written as

\[ X_t = X_0 + \int_0^t b(t,r,X_r)dr + \int_0^t \sigma(t,r,X_r) \cdot dW_r, \]

where \(X_0 \in \mathbb{R}^d\), \(W\) is a Brownian motion in \(\mathbb{R}^n\) on a given filtered probability space \((\Omega, \mathcal{F}, (\mathcal{F}_t)_{t \geq 0}, \mathbb{P})\), and \(b : \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R}^d \to \mathbb{R}^d\) and \(\sigma : \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R}^d \to \mathbb{R}^d\) satisfy the following assumptions:

**Assumption A.1.** The processes \(b\) and \(\sigma\) are adapted to \((\mathcal{F}_t)_{t \geq 0}\), and the derivatives \(\partial_t b\) and \(\partial_t \sigma\) exist. Furthermore, for \(\varphi \in \{b, \sigma, \partial_t b, \partial_t \sigma\}\), \(|\varphi(t,s,\omega)| \leq C(1 + \|\omega\|_{\mathbb{P}}^p)\) for some \(C,a > 0\).

Here, \(\|\omega\|_T := \sup_{0 \leq t \leq T} |\omega_t|\) denotes the supremum norm on the interval \([0,T]\). Saying that \(\varphi\) is adapted here is equivalent to the fact that it can be written as \(\varphi(t,r,X_r) = \varphi(t,r,X_{r\wedge}.)\). This assumption ensures that the system (21) is well defined in the following sense:

**Assumption A.2.** The SDE (21) admits a weak solution and \(\mathbb{E}\left[\sup_{t \in [0,T]} |X_t|^p\right]\) is finite for any \(p \geq 1\).

We follow [22] with this assumption. In particular, we do not require strong solutions, as the noise \(W\) is not observable, and only the process \(X\) is (or at least some components thereof; when \(X = (S,V)\) as in the present paper, only \(S\) is indeed observable). We refer the reader to [12, 13] for precise conditions on the coefficients \(b\) and \(\sigma\) ensuring weak existence of a solution. The second condition on the moment is more technical and needed for the functional Itô formula below in Theorem 3.8. Most interesting models in the finance literature satisfy these assumptions, and we refer to [22, Appendix] for sufficient conditions ensuring Assumption [X.1(ii)], in particular for the class of rough affine models developed in [2]. The key differences between (21) and a classical stochastic differential equation is (a) that both the drift and the diffusion coefficient depend on two time variables (thus violating the usual flow property), and (b) that they depend on space, not at a single instant, but evaluated along a path. The other classical issue is that the coefficients may blow up, as for the Riemann-Liouville fractional Brownian motion \(\int_0^t (t - s)^{H-1/2} dW_s\), where the power-law kernel explodes on the diagonal whenever the Hurst exponent \(H\) lies in \((0, \frac{1}{2})\). Following the terminology introduced in [22], two cases have to be distinguished:

**Assumption A.3.**

(i) (Regular case) For any \(s \in [0,T]\), \(\partial_t b(t,s,\cdot)\) and \(\partial_t \sigma(t,s,\cdot)\) exist on \([s,T]\), and for \(\varphi \in \{b, \sigma, \partial_t b, \partial_t \sigma\}\),

\[ |\varphi(t,s,\omega)| \leq C(1 + \|\omega\|_{\mathbb{P}}^p), \quad \text{for some } a,C > 0; \]
ii) (Singular case) Let $\varphi \in \{b, \sigma\}$. For any $s \in [0, T]$, $\partial_t \varphi(t, s, \cdot)$ exists on $(s, T]$, and there exists $h \in (0, \frac{1}{2})$ such that, for some $a, C > 0$,

$$|\varphi(t, s, \omega)| \leq C (1 + \|\omega\|_H^2) (t - s)^{h - 1/2} \quad \text{and} \quad |\partial_t \varphi(t, s, \omega)| \leq C (1 + \|\omega\|_H^2) (t - s)^{h - 3/2}.$$ 

The first case mainly deals with the path dependence and the absence of the Markov property, while the second one allows us to treat the presence of two time variables in the kernel, which occurs in fractional models, and in particular in the setting of Section 2 above. For any $0 \leq t \leq u$, we can decompose (21) as

$$X_u = X_0 + \int_0^t b(u, r, X_{r^-}) \, dr + \int_0^t \sigma(u, r, X_{r^-}) \, dW_r + \int_t^u b(s, r, X_{r^-}) \, dr + \int_t^u \sigma(s, r, X_{r^-}) \, dW_r.$$  

We further recall (from [22]) the concatenation notation of the paths $X$ and $\Theta^t$ before and after time $t$,

$$\langle X \Theta^t \rangle_u := X_u 1_{\{0 < u < t\}} + \Theta^t_u 1_{\{t < u < T\}}, \quad \text{for any } u, t \in [0, T].$$

A.2. Functional Itô calculus. For any $t \in [0, T]$, let $D_t$ and $C_t$ denote respectively the space of càdlàg functions on $[t, T]$ and that of continuous functions on $[t, T]$, as well as

$$\Lambda := \{(t, \omega) \in [0, T] \times D_0 : \omega_{[t,T]} \in C_t\} \quad \text{and} \quad \overline{\Lambda} := \{0, T\} \times C([0, T], \mathbb{R}^d),$$

where $\omega_{[t,T]}$ refers to the truncation of the path $\omega$ to the interval $[t, T]$. We denote by $C(\overline{\Lambda})$ the space of all functions on $\overline{\Lambda}$, continuous with respect to the distance function $d((t, \omega), (t', \omega')) := |t - t'| + \|\omega - \omega'\|_T$. For a given $u \in C(\overline{\Lambda})$, we define its (right) time derivative as

$$\partial_t u(t, \omega) := \lim_{\varepsilon \downarrow 0} \frac{u(t + \varepsilon, \omega) - u(t, \omega)}{\varepsilon}, \quad \text{for all } (t, \omega) \in \overline{\Lambda}.$$ 

Following [22], we can then define first- and second-order spatial derivatives of $u \in C(\overline{\Lambda})$ as linear and bilinear operators on $C_t$:

**Definition A.4.** The spatial derivatives of $u \in C(\overline{\Lambda})$ are defined as Gâteaux derivatives. For any $(t, \omega) \in \overline{\Lambda},$

$$\langle \partial_\omega u(t, \omega), \eta \rangle := \lim_{\varepsilon \downarrow 0} \frac{u(t, \omega + \varepsilon \eta_{[t,T]}) - u(t, \omega)}{\varepsilon}, \quad \text{for any } \eta \in C_t,$$

$$\langle \partial_\omega^2 u(t, \omega), (\eta, \zeta) \rangle := \lim_{\varepsilon \downarrow 0} \frac{\langle \partial_\omega u(t, \omega + \varepsilon \eta_{[t,T]}), \zeta \rangle - \langle \partial_\omega u(t, \omega), \zeta \rangle}{\varepsilon}, \quad \text{for any } \eta, \zeta \in C_t.$$ 

This definition of the spatial derivative in the direction $\eta \in C_t$ is obviously equivalent to

$$\langle \partial_\omega u(t, \omega), \eta \rangle = \left. \frac{du}{d\varepsilon} \right|_{\varepsilon=0} u(t, \omega + \varepsilon \eta_{[t,T]}) \big|_{\varepsilon=0}. $$

This definition is consistent with that of Dupire [13], as the perturbation acts on the time interval $[t, T]$, but not on $[0, t]$, and the distance function $d(\cdot)$ is similar to Dupire’s pseudo-distance (see also [17]). We shall further need the following two spaces:

$$C^{1,2}(\overline{\Lambda}) := \{u \in C(\overline{\Lambda}) : \varphi \in C(\overline{\Lambda}) \text{ for } \varphi \in \{\partial_t u, \partial_\omega u, \partial_\omega^2 u\}\},$$

$$C^{1,2}_+(\overline{\Lambda}) := \{u \in C^{1,2}(\overline{\Lambda}) : \varphi \text{ has polynomial growth for } \varphi \in \{\partial_t u, \partial_\omega u, \partial_\omega^2 u\}\}.$$

The definition of polynomial growth here is as follows:

**Definition A.5** (Definition 3.3 in [22]). Let $u \in C(\overline{\Lambda})$ such that $\partial_\omega u$ is well defined on $\overline{\Lambda}$. The functional $\partial_\omega u$ is said to have polynomial growth if

$$\|\partial_\omega u(t, \omega), \eta\| \leq C (1 + \|\omega\|_H^2) \|\eta_{[t,T]}\|_T, \quad \text{for all } (t, \omega) \in \overline{\Lambda}, \eta \in C_t,$$

for some $C, \alpha > 0$. It is continuous if $\overline{\Lambda} \ni (t, \omega) \mapsto \langle \partial_\omega u(t, \omega), \eta \rangle$ is continuous under $d$ for every $\eta \in C$. 

Electronic copy available at: https://ssrn.com/abstract=3400035
We now recall the main result by Viens and Zhang [22, Theorem 3.10 and Theorem 3.17], extending the Itô formula to the stochastic Volterra framework, for both regular and singular cases. The issue with the singular case (Definition A3(ii)) is that the coefficients \( \mathbf{b} \) and \( \sigma \) do not belong to \( C_1 \) any longer, so that the Gâteaux derivatives in Definition A3 do not make sense any more. In order to develop an Itô formula, those need to be amended. We refer the reader to [22, Definition 3.16] for a precise definition of the space \( C^{1,2}_{+,\beta}(\Lambda) \), where \( \beta \in (0,1) \) intuitively monitors the rate of explosion on the (time) diagonal.

**Theorem A.6.** For \( t \in [0,T] \), define the process \( Z^t := X^t \Theta^t \), and denote \( \varphi^t := \varphi(\cdot, t) \) for \( \varphi \in \{ \mathbf{b}, \sigma \} \) to emphasise the time dependence of the coefficients. Under Assumption A.3, the following Itô formula holds:

\[
\begin{align*}
&\quad dU(t, Z^t) = \left( \frac{\partial_t u(t, Z^t)}{2} + \langle \partial_{\mathbf{c}} u(t, Z^t), \mathbf{b} \rangle + \frac{1}{2} \langle \partial_{\mathbf{c}}^2 u(t, Z^t), \sigma^t \rangle \right) dt + \langle \partial_{\mathbf{c}} u(t, Z^t), \sigma^t \rangle dW_t, \\
&\quad \text{(1) in the regular case (Assumption A.3(i)), whenever } u \in C^{1,2}_{+,\beta}(\Lambda); \\
&\quad \text{(2) in the singular case (Assumption A.3(ii)) for } u \in C^{1,2}_{+,\beta}(\Lambda) \text{ with } \beta + h - \frac{1}{2} > 0, \text{ where the spatial derivatives should be understood in the regularised sense:} \\
&\quad \langle \partial_{\mathbf{c}} u(t, \omega), \phi \rangle := \lim_{\delta \downarrow 0} \langle \partial_{\mathbf{c}} u(t, \omega), \phi^\delta \rangle \quad \text{and} \quad \langle \partial_{\mathbf{c}}^2 u(t, \omega), \phi, \phi \rangle := \lim_{\delta \downarrow 0} \langle \partial_{\mathbf{c}}^2 u(t, \omega), \phi^\delta, \phi^\delta \rangle,
\end{align*}
\]

with the truncated function \( \phi^\delta(t, s, \omega) := \varphi(t \vee (s + \delta), s, \omega) \).

In the theorem, we invoked the space \( C^{1,2}_{+,\beta}(\Lambda) \), which represents the space of functions \( u : \Lambda \to \mathbb{R} \) such that there exists \( v \in C^{1,2}_{+,\beta}(\overline{\Lambda}) \) for which \( v = u \) on \( \Lambda \). The derivatives are defined similarly as restrictions on \( \Lambda \).

**Appendix B. Proof of Theorem 3.2**

Since the path-dependent PDE in the theorem is linear, existence and uniqueness of the solution is well known, and we refer to [20]. We consider a self-financing portfolio \( \Pi \) consisting of the derivative \( P \) given in (S), some quantity \( \Delta \) of stock and some other derivative \( \Psi \), i.e. at any time \( t \in [0,T] \),

\[ \Pi_t = P_t - \Delta_t S_t - \gamma_t \Psi_t. \]

Following [22] (Theorem 3.9 in the regular case and Theorem 3.16 in the singular case), we can write a functional Itô formula for the option price using Lemme A.3 under the pricing measure \( Q \):

\[
\begin{align*}
&\quad dP_t = \left( \partial_t P_t + r_t S_t \partial_x P_t + \frac{l(t, S_t, V_t)}{2} \partial_x^2 P_t + \frac{\xi(V_t)}{2} \langle \partial_{\mathbf{c}}^2 P_t, (K^t, K^t) \rangle \right) dt \\
&\quad + b(V_t) \left( \partial_x P_t, K^t \right) + l(t, S_t, V_t) \rho \xi(V_t) \left( \partial_{x,\omega} P_t, K^t \right) dt \\
&\quad + l(t, S_t, V_t) \partial_x P_t \left( \varphi dB_t + \varphi dB_t \right) + \xi(V_t) \left( \partial_{x,\omega} P_t, K^t \right) dB_t \\
&\quad =: A_P dt + l(t, S_t, V_t) \partial_x P_t dW_t + \xi(V_t) \left( \partial_{x,\omega} P_t, K^t \right) dB_t, \\
&\quad \text{where} \\
&\quad A := \partial_t \psi + r_t S_t \psi + \frac{l(t, S_t, V_t)}{2} \partial_x^2 \psi + \frac{\xi(V_t)}{2} \langle \partial_{\mathbf{c}}^2 \psi, (K^t, K^t) \rangle + b(V_t) \langle \partial_x \psi, K^t \rangle + l(t, S_t, V_t) \rho \xi(V_t) \langle \partial_{x,\omega} \psi, K^t \rangle.
\end{align*}
\]

This yields, similarly, for the portfolio \( \Pi \), under \( Q \),

\[
\begin{align*}
&\quad d\Pi_t = dP_t + \Delta_t dS_t + \gamma_t d\Psi_t \\
&\quad = A_P dt + l(t, S_t, V_t) \partial_x P_t dW_t + \xi(V_t) \left( \partial_{x,\omega} P_t, K^t \right) dB_t \\
&\quad - \gamma_t \left( A_{\Psi} dt + l(t, S_t, V_t) \partial_x \Psi_t dW_t + \xi(V_t) \left( \partial_{x,\omega} \Psi_t, K^t \right) dB_t \right) - \Delta_t \left( \mu_t dt + l(t, S_t, V_t) dW_t \right).
\end{align*}
\]

The portfolio is risk free if \( d\Pi_t = r \Pi_t dt \) and the random noise is cancelled, meaning that

\[
\begin{align*}
\begin{cases}
(24) &\quad \left( A_{\Psi} dt + l(t, S_t, V_t) \partial_x \Psi_t dW_t + \xi(V_t) \left( \partial_{x,\omega} \Psi_t, K^t \right) dB_t \right) - \Delta_t \left( \mu_t dt + l(t, S_t, V_t) dW_t \right), \\
&\quad \partial_t P_t - \gamma_t \partial_x \Psi_t - \Delta_t \partial_x S_t = 0, \\
&\quad \partial_x P_t - \gamma_t \partial_{x,\omega} \Psi_t = 0.
\end{cases}
\end{align*}
\]
since both functions $l(\cdot,\cdot,\cdot)$ and $\xi(\cdot)$ are nowhere null. The last two equalities yield

$$
\gamma_t = \frac{\partial_x P_t}{\partial_x \Psi_t} K_t^t \quad \text{and} \quad \Delta_t = \partial_x P_t - \frac{\partial_x P_t}{\partial_x \Psi_t} \partial_x \Psi_t.
$$

We can now rewrite the first equality in (25) as

$$
\left( AP_t - \gamma_t A \Psi_t - \Delta_t r_t S_t \right) dt = r_t \left( P_t - \Delta_t S_t - \gamma_t \Psi_t \right) dt,
$$

which is equivalent to

$$
(A - r_t) P_t = \left( A - r_t \right) \left( \Psi_t \right) \frac{\partial_x P_t}{\partial_x \Psi_t} K_t^t.
$$

The left-hand side is a function of $P$ only, whereas the right-hand side only depends on $\Psi$. Therefore, the only way for this equality to hold is for both sides to be equal to some function $-b$ that depends on $S_t$, $\Theta^t$ and $t$, but not on $P$ nor $\Psi$. The pricing equation for the price function is therefore

$$
(A - r_t) P_t = -\left( \partial_x P_t, K_t^t \right) \hat{b}_t.
$$

Following similar computations in classical (Markovian) stochastic volatility models, we consider $\hat{b}_t$ of the form $\hat{b}_t = b(V_t) - \xi(V_t) \lambda_t$, where $\lambda_t$ is called the market price of risk. The final pricing PDE is therefore

$$
\partial_t + r_t S_t \partial_x + \frac{l(t, S_t, V_t)^2}{2} \partial_x^2 + \frac{\xi(V_t)^2}{2} \left( \partial_{x^2} \left( K_t^t, K_t^t \right) \right) + b(V_t) \left( \partial_x K_t^t \right) + l(t, S_t, V_t) \rho \xi(V_t) \left( \partial_{x, \omega} K_t^t \right) - r_t + \left( \partial_x \omega, K_t^t \right) \hat{b}_t = 0.
$$

With a slight abuse of notations, writing $b$ in place of $\hat{b}$ proves the statement.

APPENDIX C. FURTHER NUMERICAL RESULTS

We gather in this appendix further numerical results for the rough local stochastic volatility model in Section 4.2. The first two lines correspond to prices generated by Monte Carlo and by the Deep Learning (DL) algorithm. The third line is the SSVI implied volatility given as input in (20), and lines 4 and 5 show the Monte Carlo errors and DL errors (in implied volatilities).

| Maturity: 1 week   | Strike | 0.95 | 0.98 | 1.02 | 1.05 |
|--------------------|--------|------|------|------|------|
| MC Price           | 15.09  | 15.02| 14.96| 14.94|      |
| DL Price           | 15.12  | 15.06| 14.99| 14.91|      |
| SSVI               | 15.08  | 15.03| 14.98| 14.93|      |
| DL vol error       | 0.05   | 0.03 | 0.02 | 0.02 |      |
| MC vol error       | 0.01   | 0.01 | 0.02 | 0.02 |      |

| Maturity: 1 month  | Strike | 0.89 | 0.93 | 0.98 | 1.02 | 1.07 | 1.11 |
|--------------------|--------|------|------|------|------|------|------|
| MC Price           | 15.18  | 15.11| 15.05| 14.99| 14.92| 14.85|      |
| DL Price           | 15.29  | 15. | 15.04| 14.98| 14.90| 14.84|      |
| SSVI               | 15.18  | 15.10| 15.03| 14.97| 14.91| 14.85|      |
| DL vol error       | 0.11   | 0.01 | 0.01 | 0.02 | 0.00 | 0.01 |      |
| MC vol error       | 0.00   | 0.01 | 0.02 | 0.02 | 0.01 | 0.01 |      |

| Maturity: 6 months | Strike | 0.77 | 0.83 | 0.9  | 0.97 | 1.00 | 1.1  | 1.17 | 1.23 |
|--------------------|--------|------|------|------|------|------|------|------|------|
| MC Price           | 15.37  | 15.24| 15.16| 15.03| 14.93| 14.85| 14.76| 14.72|      |
| DL Price           | 15.43  | 15.28| 15.16| 15.06| 14.95| 14.89| 14.82| 14.74|      |
| SSVI               | 15.41  | 15.27| 15.16| 15.05| 14.95| 14.86| 14.78| 14.71|      |
| DL vol error       | 0.03   | 0.01 | 0.00 | 0.01 | 0.01 | 0.03 | 0.04 | 0.03 |      |
| MC vol error       | 0.03   | 0.04 | 0.01 | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 |      |

Electronic copy available at: https://ssrn.com/abstract=3400035
### Maturity: 1 year

| Strike   | 0.72 | 0.83 | 0.94 | 1.06 | 1.17 | 1.28 | 1.39 | 1.5  |
|----------|------|------|------|------|------|------|------|------|
| MC Price | 15.49| 15.29| 15.11| 14.96| 14.81| 14.67| 14.55| 14.44|
| DL Price | 15.50| 15.31| 15.12| 14.89| 14.83| 14.71| 14.61| 14.53|
| SSVI     | 15.49| 15.27| 15.08| 14.92| 14.79| 14.67| 14.57| 14.48|
| DL vol error | 0.02 | 0.04 | 0.04 | 0.03 | 0.04 | 0.04 | 0.04 | 0.05 |
| MC vol error | 0.00 | 0.03 | 0.03 | 0.04 | 0.02 | 0.00 | 0.02 | 0.03 |

### Maturity: 2 years

| Strike   | 0.72 | 0.83 | 0.94 | 1.06 | 1.17 | 1.28 | 1.39 | 1.5  |
|----------|------|------|------|------|------|------|------|------|
| MC Price | 15.49| 15.26| 15.10| 14.98| 14.84| 14.72| 14.60| 14.51|
| DL Price | 15.51| 15.34| 15.17| 15.02| 14.89| 14.78| 14.72| 14.60|
| SSVI     | 15.46| 15.25| 15.08| 14.93| 14.80| 14.69| 14.59| 14.51|
| DL vol error | 0.05 | 0.09 | 0.09 | 0.09 | 0.09 | 0.13 | 0.10 | 0.10 |
| MC vol error | 0.03 | 0.01 | 0.02 | 0.05 | 0.04 | 0.03 | 0.01 | 0.01 |

### Maturity: 3 years

| Strike   | 0.72 | 0.83 | 0.94 | 1.06 | 1.17 | 1.28 | 1.39 | 1.5  |
|----------|------|------|------|------|------|------|------|------|
| MC Price | 15.43| 15.24| 15.09| 14.95| 14.82| 14.71| 14.60| 14.51|
| DL Price | 15.54| 15.35| 15.19| 15.05| 14.94| 14.91| 14.72| 14.64|
| SSVI     | 15.43| 15.24| 15.07| 14.93| 14.81| 14.70| 14.61| 14.53|
| DL vol error | 0.11 | 0.11 | 0.12 | 0.12 | 0.13 | 0.21 | 0.11 | 0.11 |
| MC vol error | 0.00 | 0.01 | 0.01 | 0.02 | 0.01 | 0.00 | 0.01 | 0.02 |

### References

[1] E. Abi Jaber. Lifting the Heston model. To appear in *Quantitative Finance*.
[2] E. Abi Jaber, M. Larsson, S. Pulido. Affine Volterra processes. To appear in *Annals of Applied Probability*.
[3] E. Alós, J. León and J. Vives. On the short-time behavior of the implied volatility for jump-diffusion models with stochastic volatility. *Finance and Stochastics*, 11(4), 571-589, 2007.
[4] J. Ba and D. Kingma. Adam: a method for stochastic optimization. *Proceedings of the International Conference on Learning Representations*, May 2015.
[5] G. Barles and P.E. Souganidis. Convergence of approximation schemes for fully nonlinear second-order equation. *Asymptotic Analysis*, 4: 271-283, 1991.
[6] C. Bayer, C. Ben Hammouda and R. Tempone. Hierarchical adaptive sparse grids for option pricing under the rough Bergomi model. *arXiv:1812.08533*, 2018.
[7] C. Bayer, P. Friz, P. Gassiat, J. Martin and B. Stemper. A regularity structure for rough volatility. *Quantitative Finance*, 16(6): 1-18, 2015.
[8] C. Bayer, P. Friz and J. Gatheral. Pricing under rough volatility. *Quantitative Finance*, 16(6): 1-18, 2015.
[9] C. Bayer, P. Friz, A. Gulisashvili, B. Horvath and B. Stemper. Short-time near the money skew in rough fractional stochastic volatility models. To appear in *Quantitative Finance*.
[10] M. Bennedsen, A. Lunde and M.S. Pakkanen. Hybrid scheme for Brownian semistationary processes. *Finance and Stochastics*, 21(4): 931-965, 2017.
[11] M. Bennedsen, A. Lunde and M.S. Pakkanen. Decoupling the short- and long-term behavior of stochastic volatility. *arXiv:1610.05672*, 2017.
[12] M.A. Berger and V.J. Mizel. Volterra Equations with Itô Integrals, I. *Journal of Integral Equations*, 2(3): 187-245, 1980.
[13] M.A. Berger and V.J. Mizel. Volterra Equations with Itô Integrals, II. *Journal of Integral Equations*, 2(4): 319-337, 1980.
[14] F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journ. Political Economy*, 81(3): 637-654, 1973.
[15] F. Comte and E. Renault. Long memory continuous time models. *Journal of Econometrics*, 73(1): 101-149, 1996.
[16] I. Ekren, N. Touzi and J. Zhang. On viscosity solutions of path-dependent PDEs. *Annals Prob*., 42(1): 204-236, 2014.
[17] I. Ekren, N. Touzi and J. Zhang. Viscosity solutions of fully nonlinear parabolic path-dependent PDEs: Part I. *Annals of Probability*, 44(2): 1212-1253, 2016.
[18] I. Ekren, N. Touzi and J. Zhang. Viscosity solutions of fully nonlinear parabolic path-dependent PDEs: Part II. *Annals of Probability*, 44(4): 2507-2553, 2016.
[67] Z. Ren, N. Touzi and J. Zhang, An overview of viscosity solutions of path-dependent PDEs. *Stochastic Analysis and Applications*. Springer Proceedings in Mathematics and Statistics, 100: 397-454, 2014.

[68] M. Romano and N. Touzi. Contingent claims and market completeness in a stochastic volatility model. *Mathematical Finance*, 7(4): 399-412, 1997.

[69] M. Sabate Vidales, D. Šiška and L. Szpruch. Martingale functional control variates via deep learning. *arXiv:1810.05099*, 2018.

[70] J. Sirignano and K. Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375: 1339-1364, 2018.

[71] H. Stone. Calibrating rough volatility models: a convolutional neural network approach. *arXiv:1812.03637*, 2018.

[72] F. Viens and J. Zhang. A martingale approach for fractional Brownian motions and related path-dependent PDEs. *arXiv:1712.03637*, 2018.

[73] J. Zhang and J. Zhuo. Monotone schemes for fully nonlinear parabolic path dependent PDEs. *Journal of Financial Engineering*, 1, 2014.

**Department of Mathematics, Imperial College London, and Alan Turing Institute**

*E-mail address*: a.jacquier@imperial.ac.uk

**Lloyds Banking Group plc, Commercial Banking, 10 Gresham Street, London, EC2V 7AE, UK**

*E-mail address*: Mugad.Oumgari@lloydsbanking.com

Electronic copy available at: https://ssrn.com/abstract=3400035