Robust pricing and hedging via neural SDEs

David Šiška[1,2]

Joint work with: Patryk Gierjatowicz[1], Marc Sabate-Vilades[1,3], Lukasz Szpruch[1,3] and Zan Zuric[4]

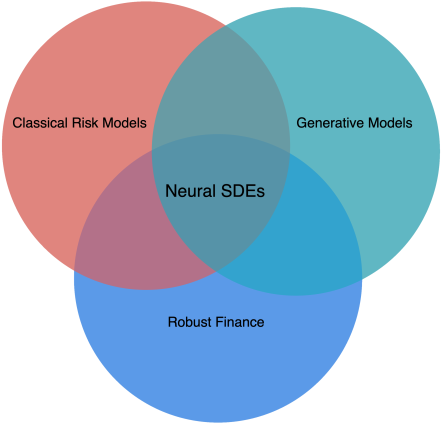
CMStatistics 2021, Deep calibration of financial models

[1]University of Edinburgh
[2]Vega Protocol
[3]The Alan Turing Institute
[4]Imperial College Lodon

Modelling in Quantitative Finance

Aim: price and hedge (exotic) derivatives consistently with available market data.

1. Classical models (think Heston, SABR or Hull–White, LMM)
2. Robust finance
3. Generative models
4. Neural SDEs



Devising classical risk models

Until recently, models in finance and economics were mostly conceived in a three step fashion:

1. statistical properties of time-series $\Longrightarrow$ key stylized facts
2. handcrafting a parsimonious model that captures those and
3. calibration and validation of the handcrafted model.

Model complexity was undesirable, overparametraziation a sign of mediociry.[5]

[5] George Box, 1976. "Since all models are wrong the scientist cannot obtain a 'correct' one by excessive elaboration. On the contrary following William of Occam he should seek an economical description of natural phenomena. Just as the ability to devise simple but evocative models is the signature of the great scientist so overelaboration and overparameterization is often the mark of mediocrity."

Classical risk models

Pros:

- Interpretable parameters
- Relatively easy to calibrate with relatively small amount of data
- Several decades of underpinning research

Cons:

- Lack of systematic framework for model selection
- Knightian uncertainty (unknown unknowns)
- Limited expressivity

Robust finance

- There are infinitely many models that are consistent with the market
- $\mathcal{M}$ - set of all martingale measures that are calibrated to data
- Compute bounds for the price i.e.

$$\sup_{\mathbb{Q}\in\mathcal{M}} \mathbb{E}^{\mathbb{Q}}[\Psi] \quad \text{and} \quad \inf_{\mathbb{Q}\in\mathcal{M}} \mathbb{E}^{\mathbb{Q}}[\Psi]$$

- Use duality theory to deduce (semi-static) hedging strategy
- The obtain bounds typically too wide to be of practical value
- Challenges:
  a) Incorporate prior information to restrict a search space $\mathcal{M}$
  b) Design efficient algorithms for computing price bounds and corresponding hedges

Generative modelling

- Generative models such as Generative Adversarial Networks or Variational Autoencoders demonstrated a great success in seemingly high dimensional setups.
- Input: Source (latent) distribution μ and target (data) distribution ν i.e input-output data
- A generative model is a transport map T from μ to ν i.e T is a map that "pushes μ onto ν". We write $T_{\#}\mu = \nu$.
- Parametrise transport map $T(\theta)$, $\theta \in \mathbb{R}^p$, e.g some network architecture or Heston model
- Seek $\theta^\star$ s.t $T(\theta^\star)_{\#}\mu \approx \nu$.
- Need to make the choice of the metric

$$D(T(\theta)_{\#}\mu, \nu) := \sup_{f \in \mathcal{K}} \| \int f(x)(T(\theta)_{\#}\mu)(dx) - \int f(x)\nu(dx) \|$$

- $\mathcal{K}$ could be set of options we want to calibrate to, could be neural network.
- The modelling choices are
  1. metric D
  2. parametrisation of T
  3. algorithm used for training

Generative modelling with neural networks in finance

Pros:

- Expressive and work in high dimensions
- By design data driven, adaptable to change in environment
- Provide new perspective on classical problems in finance

Cons:

- Parameters are not interpretable - black box approach
- Training algorithms are data hungry
- Models might be hard to work with e.g how to go from $\mathbb{Q}$ to $\mathbb{P}$?
- A field largely empirical, lack of standardised benchmarks, lack of theoretical guarantees

Read: [Kondratyev and Schwarz, 2019, Buehler et al., 2020b, Henry-Labordere, 2019, Xu et al., 2020, Ni et al., 2020, Wiese et al., 2020, Buehler et al., 2020a]

Robust pricing and hedging via neural SDEs

Neural SDEs

- We build an Itô process $(X_t^\theta)_{t\in[0,T]}$, with parameters $\theta \in \mathbb{R}^p$

$$
\begin{aligned}
dS_t^\theta &= rS_t^\theta\,dt + \sigma^S(t, X_t^\theta, \theta)\,dW_t\,,\\
dV_t^\theta &= b^V(t, X_t^\theta, \theta)\,dt + \sigma^V(t, X_t^\theta, \theta)\,dW_t\,,\\
X_t^\theta &= (S_t^\theta, V_t^\theta)\,,
\end{aligned}
$$

  where σ^S, b^V, σ^V are given by neural networks.
- The model induces a martingale probability measure $\mathbb{Q}(\theta)$.
- Solution map is an instance of causal transport.
- Neural SDEs are easy to work with e.g consistent change from $\mathbb{Q}$ to $\mathbb{P}$.
- Neural SDEs mitigate, to some extent, the concerns that regulators have around use of black-box solutions to manage financial risk.

Neural SDEs

i) **Calibration to market prices** Find model parameters θ^* such that model prices match market prices:

$$\theta^* \in \arg\min_{\theta \in \Theta} \sum_{i=1}^{M} \ell(\mathbb{E}^{\mathbb{Q}(\theta)}[\Phi_i], \mathfrak{p}(\Phi_i)) .$$

ii) **Robust pricing** Find model parameters $\theta^{l,*}$ and $\theta^{u,*}$ which provide robust arbitrage-free price bounds for an illiquid derivative, subject to available market data:

$$\theta^{l,*} \in \arg\min_{\theta \in \Theta} \mathbb{E}^{\mathbb{Q}(\theta)}[\Psi] , \quad \text{subject to} \quad \sum_{i=1}^{M} \ell(\mathbb{E}^{\mathbb{Q}(\theta)}[\Phi_i], \mathfrak{p}(\Phi_i)) = 0 ,$$

$$\theta^{u,*} \in \arg\max_{\theta \in \Theta} \mathbb{E}^{\mathbb{Q}(\theta)}[\Psi], \quad \text{subject to} \quad \sum_{i=1}^{M} \ell(\mathbb{E}^{\mathbb{Q}(\theta)}[\Phi_i], \mathfrak{p}(\Phi_i)) = 0 .$$

Here ℓ is a convex loss function s. t. $\ell(x, y) = 0$ iff $x = y$.

Stochastic Optimisation

Let $M = 1$ and

$$h(\theta) = \ell\Big(\mathbb{E}^{\mathbb{Q}(\theta)}[\Phi], \mathfrak{p}(\Phi)\Big)\,.$$

Then in the gradient step update we have

$$\partial_\theta h(\theta) = \partial_x \ell\left(\mathbb{E}^{\mathbb{Q}}[\Phi(X^\theta)], \mathfrak{p}(\Phi)\right)\mathbb{E}^{\mathbb{Q}}[\partial_\theta \Phi(X^\theta)]\,.$$

Since ℓ is typically not an identity function, a mini-batch estimator of $\partial_\theta h(\theta)$, obtained by replacing $\mathbb{Q}$ with $\mathbb{Q}^N$ given by

$$\partial_\theta h^N(\theta) := \partial_x \ell\left(\mathbb{E}^{\mathbb{Q}^N}[\Phi(X^\theta)], \mathfrak{p}(\Phi)\right)\mathbb{E}^{\mathbb{Q}^N}[\partial_\theta \Phi(X^\theta)]\,,$$

is a biased estimator of $\partial_\theta h$.

Lemma 1

For $\ell(x, y) = |x - y|^2$, we have

$$\left|\mathbb{E}^{\mathbb{Q}}\left[\partial_\theta h^N(\theta)\right] - \partial_\theta h(\theta)\right| \leq \frac{2}{N}\left(\mathbb{V}ar^{\mathbb{Q}}[\Phi(X^\theta)]\right)^{1/2}\left(\mathbb{V}ar^{\mathbb{Q}}[\partial_\theta \Phi(X^\theta)]\right)^{1/2}\,.$$

Neural SDEs - Algorithm

Input: $\pi = \{t_0, t_1, \ldots, t_{N_{\text{steps}}}\}$ time grid for numerical scheme.
Input: $(\Phi_i)_{i=1}^{N_{\text{prices}}}$ option payoffs.
Input: Market option prices $\mathfrak{p}(\Phi_j)$, $j = 1, \ldots, N_{\text{prices}}$.
for epoch : 1 : N_{epochs} **do**

Generate N_{trn} paths $(x_{t_n}^{\pi,\theta,i})_{n=0}^{N_{\text{steps}}} := (s_{t_n}^{\pi,\theta,i}, v_{t_n}^{\pi,\theta,i})_{n=0}^{N_{\text{steps}}}$, $i = 1, \ldots, N_{\text{trn}}$ using Euler scheme.

During one epoch: Freeze ξ, use Adam to update θ, where

$$\theta = \widehat{\operatorname*{argmin}_{\theta}} \sum_{j=1}^{N_{\text{prices}}} \left(\mathbb{E}^{N_{\text{trn}}} \left[\Phi_j\left(X^{\pi,\theta}\right) - \sum_{k=0}^{N_{\text{steps}}-1} \bar{\mathfrak{h}}(t_k, \tilde{X}_{t_k}^{\pi,}, \xi_j) \Delta \tilde{\bar{S}}_{t_k}^{\pi,} \right] - \mathfrak{p}(\Phi_j) \right)^2$$

During one epoch: Freeze θ, use Adam to update ξ, by optimising the sample variance

$$\xi = \widehat{\operatorname*{argmin}_{\xi}} \sum_{j=1}^{N_{\text{prices}}} \mathbb{V}\text{ar}^{N_{\text{trn}}} \left[\Phi_j\left(X^{\pi,\theta}\right) - \sum_{k=0}^{N_{\text{steps}}-1} \bar{\mathfrak{h}}(t_k, X_{t_k}^{\pi,\theta}, \xi_j) \Delta \tilde{\bar{S}}_{t_k}^{\pi,\theta} \right]$$

end for
return θ, ξ_j for all prices $(\Phi_i)_{i=1}^{N_{\text{prices}}}$.

Calibration to SPX market prices I

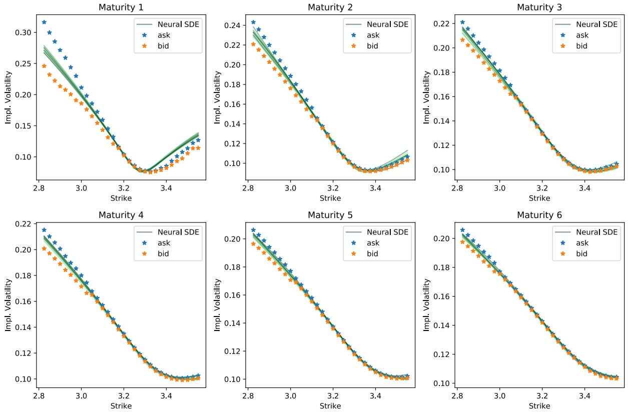


Figure: Comparing market implied volatlity to the model implied volatility for the neural SDE LSV model when targeting only the *market data*. Implied volatility curves of 10 different calibrated neural SDEs are presented in comparison to the market bid-ask spread for different maturities.

Calibration to SPX market prices II

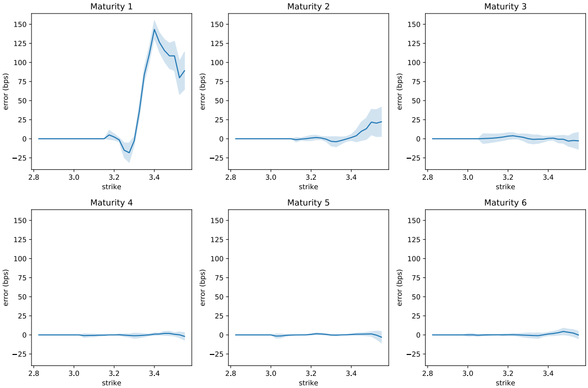


Figure: The average error and standard deviation in basis points (bps) of the implied volatility curves from Figure 1. The error is considered to be zero if the implied volatility falls into the bid-ask spread.

Robust pricing against SPX

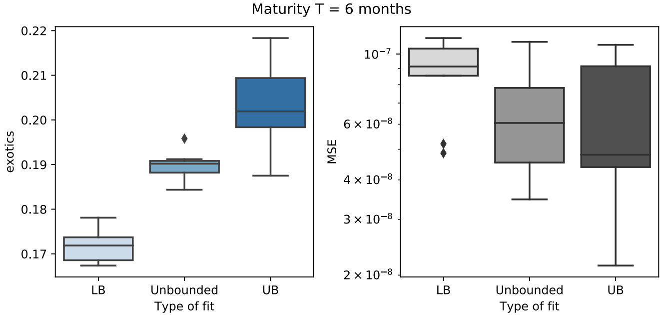


Figure: Box plots for the Local Stochastic Volatility model. Exotic option price quantiles are in blue in the left-hand box-plot groups. The MSE quantiles of market data calibration is in grey, in the right-hand box-plot groups. Each box plot comes from 10 different runs of neural SDE calibration, where in each run the parameters of the neural SDE are initialised with a different seed. The three box-plots in each group arise respectively from aiming for a lower bound of the illiquid derivative (left), only calibrating to market and then pricing the illiquid derivative (middle) and aiming for a upper bound of the illiquid derivative (right).

Calibration to SPX while minimising exotic price I

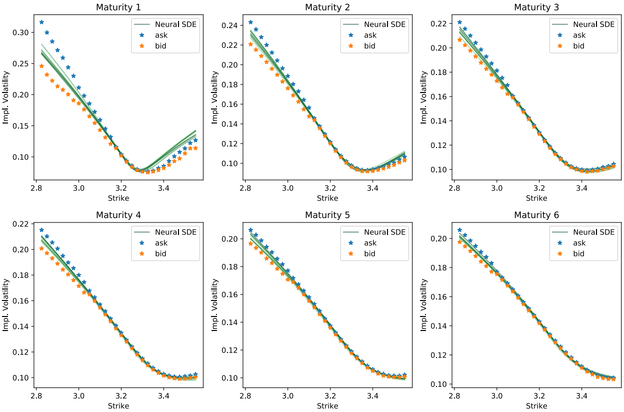


Figure: Comparing market implied volatlity and model implied volatility for the neural SDE LSV model when targeting the *market data* **and minimising the exotic option price**. We see implied volatility curves of the 10 calibrated neural SDEs vs. the market bid-ask spread for different maturities.

Calibration to SPX while minimising exotic price II

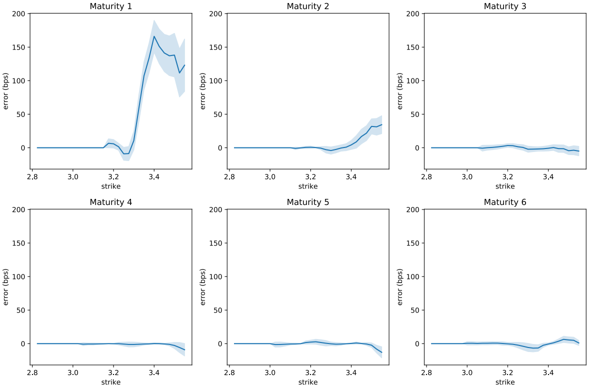


Figure: Average error and standard deviation in basis points (bps) of the implied volatility curves from targeting the *market data* **and minimising the exotic option price**. The error is considered to be 0 if the implied volatility falls into the bid-ask spread.

Control variate effect on training

Learning the “hedging strategy” and using it as control variate stabilises training:

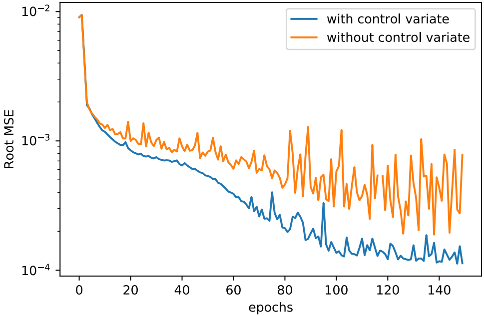


Figure: Root Mean Squared Error of calibration to Vanilla option prices with and without hedging strategy parametrisation

- See also [Cuchiero et al., 2020].

Extensions

- Neural SDE model in real-world measure $\mathbb{P}(\theta)$
- Let $\zeta : [0, T] \times \mathbb{R}^d \times \mathbb{R}^p \to \mathbb{R}^n$ be another parametric function
- Let

$$b^{S,\mathbb{P}}(t, X_t^\theta, \theta) := rS_t^\theta + \sigma^S(t, X_t^\theta, \theta)\zeta(t, X_t^\theta, \theta)\,,$$
$$b^{V,\mathbb{P}}(t, X_t^\theta, \theta) := b^V(t, X_t^\theta, \theta) + \sigma^V(t, X_t^\theta, \theta)\zeta(t, X_t^\theta, \theta)\,.$$

- We now define a real-world measure $\mathbb{P}(\theta)$ via the Radon–Nikodym derivative

$$\frac{d\mathbb{P}(\theta)}{d\mathbb{Q}(\theta)} := \exp\left(\int_0^T \zeta(t, X_t^\theta, \theta)\, dW_t + \frac{1}{2}\int_0^T |\zeta(t, X_t^\theta, \theta)|^2\, dt\right)\,.$$

- Using Girsanov theorem we can find Brownian motion $(W_t^{\mathbb{P}(\theta)})_{t\in[0,T]}$ such that

$$dS_t^\theta = b^{S,\mathbb{P}}(t, X_t^\theta, \theta)\, dt + \sigma^S(t, X_t^\theta, \theta)\, dW_t^{\mathbb{P}(\theta)}\,,$$
$$dV_t^\theta = b^{V,\mathbb{P}}(t, X_t^\theta, \theta)\, dt + \sigma^V(t, X_t^\theta, \theta)\, dW_t^{\mathbb{P}(\theta)}\,.$$

- Train with historical time series statistics (moments) or e.g. Wasserstein distances between generated and empirical measures.

Neural SDEs

Pros:

- Expressive yet consistent with classical framework
- By design data driven, adaptable to changes in environment
- Provide consistent models for calibrating under $\mathbb{Q}$ and $\mathbb{P}$
- Provide systematic framework for model selection
- Ability to learn in low data regime (SDEs are a good prior)
- (Some) Theoretical guarantees for the generalisation error

Cons:

- Individual parameters are not interpretable, but the models are.
- Computationally intensive, but recalibration is cheap.

Read:

- Market Model using NSDEs [Cohen et al., 2021],
- NSDEs with a prior on vol process [Cuchiero et al., 2020],
- NSDE in a signature feature space [Arribas et al., 2020]
- Robust pricing and hedging with NSDEs [Gierjatowicz et al., 2020].

Thank you.

References I

[Arribas et al., 2020] Arribas, I. P., Salvi, C., and Szpruch, L. (2020). Sig-sdes model for quantitative finance. *arXiv preprint arXiv:2006.00218.*

[Buehler et al., 2020a] Buehler, H., Horvath, B., Lyons, T., Perez Arribas, I., and Wood, B. (2020a). A data-driven market simulator for small data environments. *Available at SSRN 3632431.*

[Buehler et al., 2020b] Buehler, H., Horvath, B., Lyons, T., Perez Arribas, I., and Wood, B. (2020b). Generating financial markets with signatures. *Available at SSRN.*

[Cohen et al., 2021] Cohen, S. N., Reisinger, C., and Wang, S. (2021). Arbitrage-free neural-sde market models.

[Cuchiero et al., 2020] Cuchiero, C., Khosrawi, W., and Teichmann, J. (2020). A generative adversarial network approach to calibration of local stochastic volatility models. *arXiv preprint arXiv:2005.02505.*

[Gierjatowicz et al., 2020] Gierjatowicz, P., Sabate-Vidales, M., Siska, D., Szpruch, L., and Zuric, Z. (2020). Robust pricing and hedging via neural sdes. *Available at SSRN 3646241.*

[Henry-Labordere, 2019] Henry-Labordere, P. (2019). Generative models for financial data. *Available at SSRN 3408007.*

[Kondratyev and Schwarz, 2019] Kondratyev, A. and Schwarz, C. (2019). The market generator. *Available at SSRN 3384948.*

[Ni et al., 2020] Ni, H., Szpruch, L., Wiese, M., Liao, S., and Xiao, B. (2020). Conditional sig-wasserstein gans for time series generation. *arXiv preprint arXiv:2006.05421.*

References II

[Wiese et al., 2020] Wiese, M., Knobloch, R., Korn, R., and Kretschmer, P. (2020). Quant gans: Deep generation of financial time series. *Quantitative Finance*, pages 1–22.

[Xu et al., 2020] Xu, T., Wenliang, L. K., Munn, M., and Acciaio, B. (2020). Cot-gan: Generating sequential data via causal optimal transport. *arXiv preprint arXiv:2006.08571*.