A DEEP LEARNING APPROACH TO EXOTIC OPTION PRICING UNDER LSVOL

Dr Katia Babbar
Visiting Research Fellow, University of Oxford

AI Wealth Technologies

QuantBright Ltd

*Joint research with William McGhee

Why consider Deep Neural Networks for Financial Derivatives?

In the past decades:

- There has been a significant growth in the use of financial derivatives
- Models have evolved in sophistication to better capture market dynamics
- At the same time there has been an increase in demand for both:
    - speed
    - range of risk calculations

Contract Volume

Model Sophistication

Risk Calculations

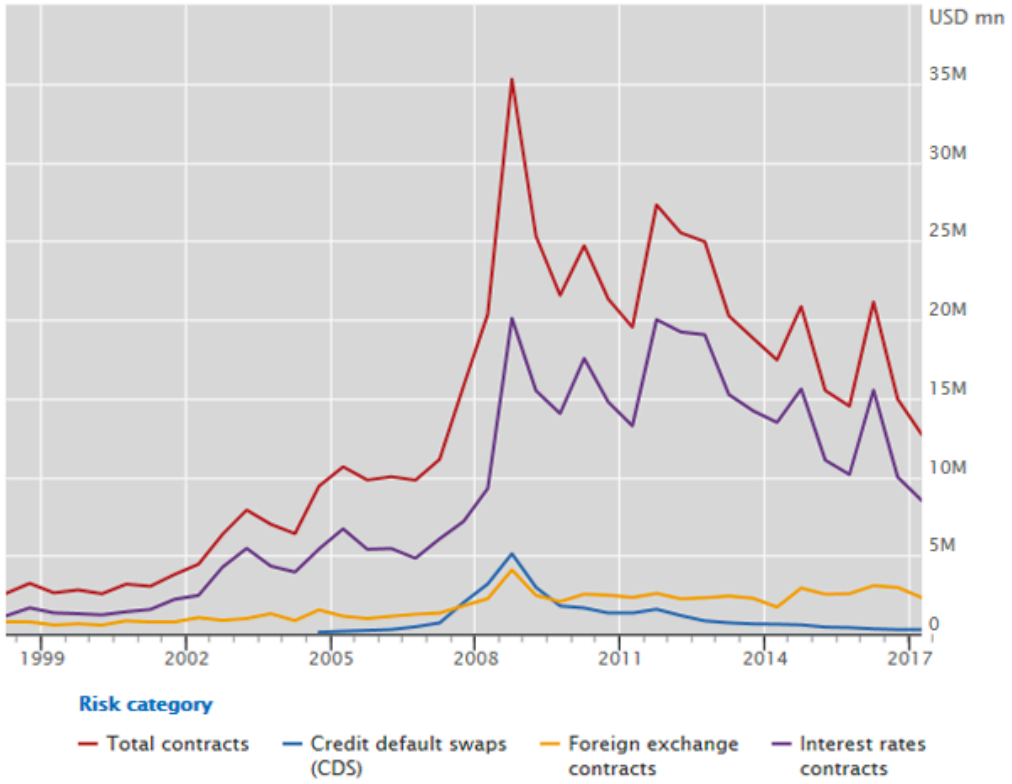
Speed of Information

Growth in use of Financial Derivatives

- Total outstanding notional
- For risk calculations, the outstanding total number of contracts for the relevant bank matters for infrastructure implications
- An estimate is to assume that there are hundreds of thousands of outstanding FX Exotic Options contracts

BIS OTC derivatives statistics

Outstanding gross market value, in trillions (ie in million millions, eg 20M equals 20 million millions, or 20 trillion) of US dollars.



Universe of models

- Speed has worked as a constraint in the choice of models used in production by banks
- Where possible, preference was skewed towards semi-analytic and asymptotic approximations, at times compromising accuracy
- In most cases, more intense numerical methods need to be employed, such as Monte Carlo, Finite Difference Schemes for PDEs, Numerical Integration…

To implement accurate models whilst retaining acceptable speed the industry has adopted use of compute grids / GPUs to process calculations in parallel (at a cost)

Models have evolved to better account for market dynamics

- Bachelier (1900): Pricing options assuming the underlying asset is a Brownian Motion
- Black-Scholes (1973) & Merton (1973): Modelling the underlying asset as a Geometric Brownian Motion (GBM) *(introduction of arbitrage-free pricing, payoff hedging replication)*
- Local Volatility Dupire (1994) & Derman-Kani (1994) Option valuation accounting for weaknesses in Black-Scholes modelling and capturing the volatility smile (asset prices don't follow a GBM)
- Stochastic Volatility Hull-White (1987) & Heston (1993): more than one factor explains dynamic of underlying asset (volatility itself is stochastic)
- Local-Stochastic Volatility (LSVol) - JP Morgan (1995) & Lipton-McGhee (2002): embed desirable dynamics from both Local Volatility and Stochastic Volatility models.

Running risk calculations for a Bank's portfolio is intense and expensive

Typical Intraday needs of a Trading Desk:

- Assume portfolio of 10,000 exotic options
- Scenarios and different risk views are generated to understand market changes on portfolio
- Average: 1,000 valuations per contract
- Each risk calculation 800ms for a production model (LSVol) using C++ Finite Difference Scheme
- It would take ~95 days on a single CPU
- Parallelise process using Compute Grids
- Infrastructure to support risk updates every 30mins requires 4-5k core production nodes
- For regulatory purposes the number of risk calculations can be significantly higher
- ~ £10s of millions in hardware/infrastructure/people cost

In the meantime: advances in (Deep) Neural Networks

Explosion in Data	Data is being captured and stored at a phenomenal pace
Increase in Computational Power with decrease in cost	Development of GPU technology (and now TPUs)
Sophistication of Algorithms	Ability to train, especially Deep, Neural Networks
Crowd Sourcing	Collaboration and outsider's view

The vast majority of applications of neural networks in the field of machine learning aim to extract modellable features from the data to allow for *prediction*. Here we will be using a fundamental property of neural networks allowing us to exploit them as high-dimensional interpolators.

Deep Neural Networks are powerful universal interpolators

The power of neural networks is their ability to represent any smooth multidimensional function.
This stems from the Universal Approximation Theorem

The Universal Approximation Theorem tells us it is possible but not whether it is computationally feasible in practice.

Neural Networks are *Universal Approximators*. Once trained they are orders of magnitude faster than Traditional numerical schemes in Quantitative Finance.

***Question**: in practice, can we train neural networks to represent complex pricing functions to the accuracy of existing numerical methods in finance?*

Theorem: Let $\phi(.)$ be a non-constant, bounded and monotonically increasing continuous function. Let $V_m \in \mathbb{R}^m$ be any compact subset of $\mathbb{R}^m$. The space of continuous functions on V_m is denoted by $C(V_m)$. Given $\epsilon > 0$ and any function $f \in C(V_m)$, there exists an integer N and constants $\nu_i, b_i \in \mathbb{R}$ and real vectors $\omega_i \in \mathbb{R}^m$, where $i = 1, ..., N$ such that

$$F(x) = \sum_{i=1}^{N} \nu_i \phi(\omega_i^T x + b_i) \quad (1)$$

as an approximate realisation of the function f, which is independent of $\phi(.)$. Specifically,

$$|F(x) - f(x)| < \epsilon \quad (2)$$

for all $x \in V_m$.

Cybenko (1989), Hornik (1990)

Deep Neural Networks open new possibilities in Finance

- DNNs as a fast numerical method **(This talk's focus!)**
  - The extensive model investment in banks which may span over 20 years is preserved, with deep learning models presenting a potentially novel numerical approach which is vastly superior in performance
- DNNs broaden the universe of models
  - Given their ability to be trained off-line, consider even more sophisticated models which have the potential to better describe realistic market dynamics to be implemented in a production like environment
- DNNs used for prediction of Market Dynamics
  - Use financial data and/or other data (media, news, …) to predict price moves
  - Use of data to infer a data-driven risk-neutral model (…)

Deep Neural Networks application to Options pricing: a few key papers in this context

Black-Scholes models:

- Hutchison, Lo & Poggio (1994) "A Nonparametric Approach to Pricing and Hedging Derivatives Securities [...]"
- Morelli et al (2004) "Pricing Financial Derivatives with Neural Networks"
- Brostrom & Kristiansson (2018) "Exotic Derivatives and Deep Learning"
- Ferguson & Green (2018) "Deep Learning Derivatives"

Stochastic Volatility Models - matching vanilla prices:

- McGhee (2018) "An Artificial Neural Network Representation of the **SABR** Stochastic Volatility Model" x 10,000 faster
- Horvath, Muguruza & Tomas (2019) "Deep Learning Volatility" – ***SABR, Heston and beyond... Rough Vol models.***
- Liu, Oosterlee & Bohte (2019) "Pricing Options and computing implied volatilities using neural networks"

Deep Neural Networks application to Options pricing: a few key papers in this context

Direct use of Neural Networks for Calibration:

- Hernandez (2017) "Model calibration with neural networks"

Models beyond Black-Scholes for Exotic Options:

- De Spiegeleer, Madan, Reyners & Schoutens (2018) "Machine Learning for Quantitative Finance: Fast Derivative Pricing, Hedging and Fitting"– ***Use of GPR for Heston…. But: very limited training region, only negative correlations?***

Exotic Options Deep Learning with Local Stochastic Volatility

- Babbar & McGhee (2019) "A Deep Learning Approach to Exotic Option Pricing under LSVol" – *this presentation, paper in preparation.*

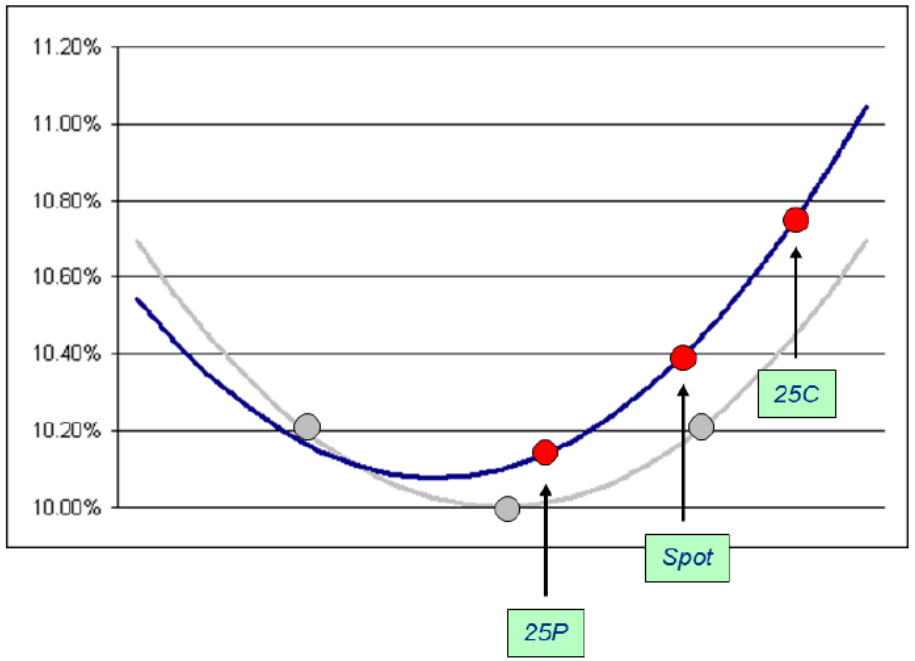
Deep Neural Networks for Exotic Option pricing beyond Black-Scholes

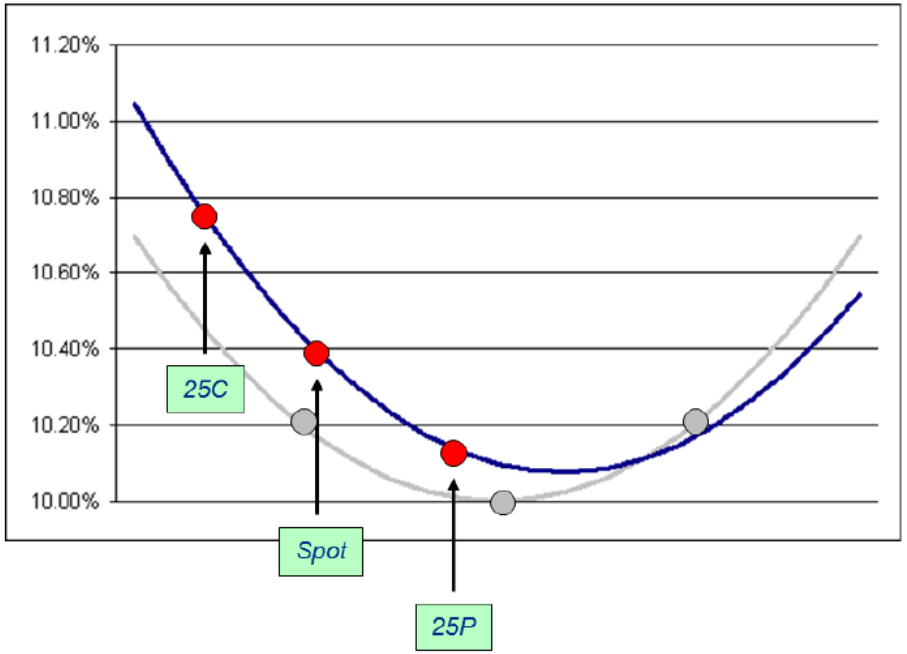
- (Deep) neural networks have been applied to try and represent pricing of derivatives using Black-Scholes assumptions on the underlying asset(s).
- They have also been employed in the context of enriching model assumptions (Buhler et al).
- McGhee (2018), Horvath et al (2019) & Lieu (2019) have been the first to apply deep neural networks for vanilla pricing under stochastic volatility models and used in the context of pricing vanillas. The speed up is particularly relevant in the context of calibration where otherwise there are unsatisfactory solutions (SABR in the wings..).
- In relation to Exotic Option pricing, the contracts considered by previous literature were (mostly) **not path-dependent,** with some exceptions e.g. De Spiegeleer et al.
- This is the first example of exploring the application of deep learning models for pricing of **path-dependent** Exotic Options using standard market models (**LSVol**), covering a realistic range of parameters.

LSVol intuition

- A local volatility model is high spot/risk reversal (skew) correlation, whereas large spot moves are associated with large surface moves
- In blending local with stochastic volatility the spot/risk reversal correlation is reduced in return for greater degree of vol-of-vol
- In a local volatility model there is a unique surface at each future spot level S(t) (with a large skew)
- With increasing vol-of-vol a distribution of surfaces is introduced each with reduced skew
- As vol-of-vol increases, the expected volatility surface skew decreases and strangle levels increase

Local vol model





At current spot:

- *no risk reversal*
- *ATM vol of 10%*

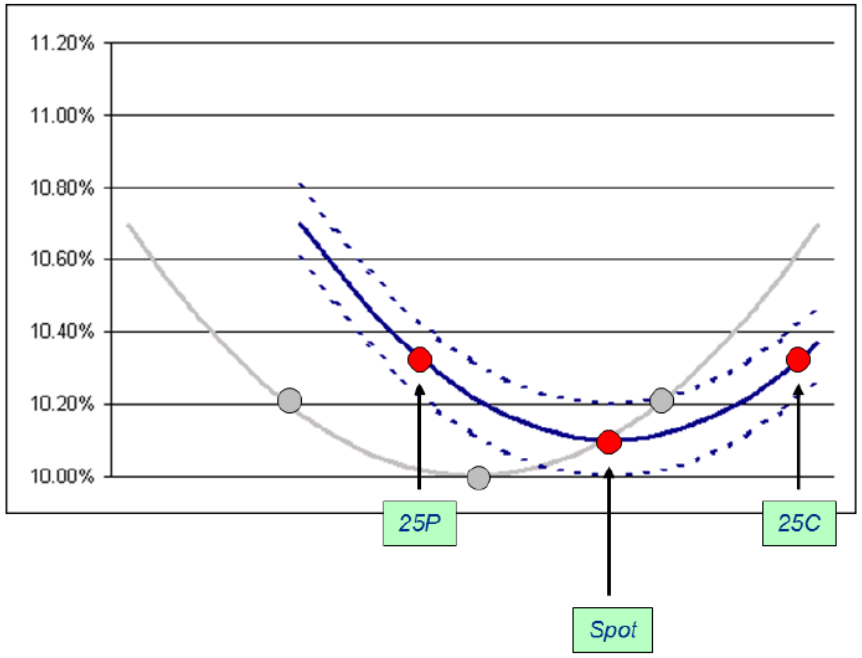
For higher spot:

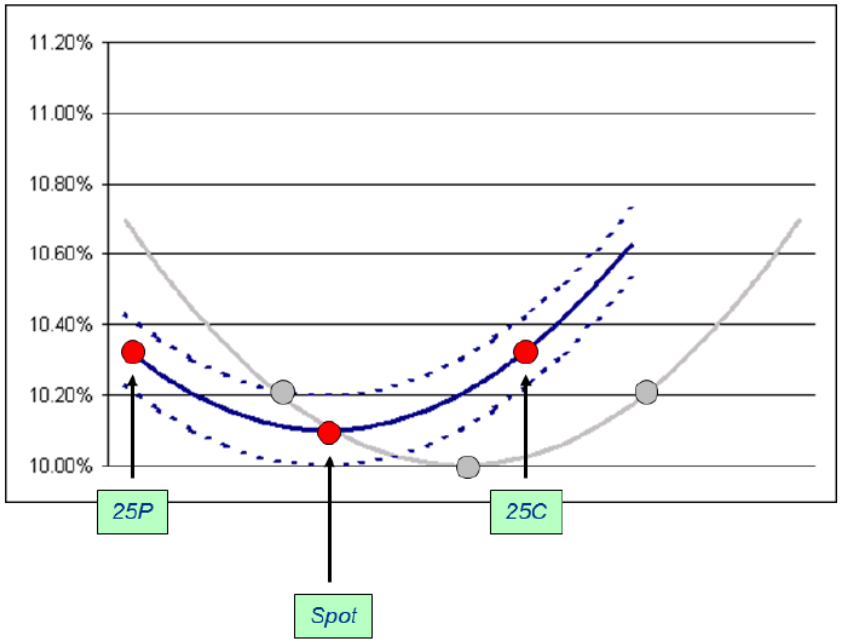
- *risk reversal bid for upside*
- *ATM vol of rises*

For lower spot:

- *risk reversal bid for dnside*
- *ATM vol of rises*

Stochastic Vol model (zero correlation)





At current spot:

- *no risk reversal*
- *ATM vol of 10%*

For higher spot:

- ***no*** *risk reversal*
- *ATM vol rises but uncertain*

For lower spot:

- ***no*** *risk reversal*
- *ATM vol rises but uncertain*

The LSVol model used to value Exotic Options

- LSVol model:

$$dS_t = \mu S_t dt + \lambda(t, S_t)\sigma(t) S_t dW_S(t) \qquad d < W_S, W_\sigma >_t = \rho dt$$
$$d\sigma_t = \omega \xi \sigma_t dW_\sigma(t)$$

- Solution to the following underlying PDE (transforming to log-coordinates X,Y):

$$\frac{\partial V}{\partial t} + \left(\mu - \frac{1}{2}\lambda(e^{X(t)}, t)^2 e^{2Y(t)}\right)\frac{\partial V}{\partial X} + \frac{1}{2}\lambda(e^{X(t)}, t)^2 e^{2Y(t)}\frac{\partial^2 V}{\partial X^2}$$
$$-\frac{1}{2}\xi^2\frac{\partial V}{\partial Y} + \frac{1}{2}\xi^2\frac{\partial^2 V}{\partial Y^2} + \rho\lambda(e^{X(t)}, t)e^{Y(t)}\xi\frac{\partial^2 V}{\partial X \partial Y} - r_d V = 0$$

- Solved the PDE on a 2-factor ADI Finite Difference Scheme (e.g. time, spot, vol: 200x500x50 points)
- C++ implementation using a 3.4GHz CPU with production level accuracy leads to 800ms per valuation
- For the purposes of this research, parameters are kept constant (see later discussion)

Volatility surface construction

- In general, where the volatility surface is generated using SABR, it will be the *SABR Approximation* function that is used
- This is inconsistent with the pure SABR model on 2FPDE (SABR approximation parameters deviate from SABR 2FPDE parameters) – but used for practical performance reasons. Practitioners have a challenge.
- In McGhee (2018) an ANN is constructed which is consistent with the accurate 2FPDE and performs 10,000x faster than using the 2FPDE
- In this project, the SABR ANN is used to construct the SABR volatility surface from which the local volatility surface is derived
- The SABR parameters in the volatility surface are consistent with those of the 2FPDE
- The pure Stochastic Volatility version of the LSVol model is therefore consistent with pure SABR

LSVol method

1. Choose SABR parameters (σ_0, ξ, ρ)
2. Generate volatility surface (implied)
   - In practice, the volatility surface is given by the market (SIV, SABR app, Spline…)
   - Here we generate vol surface using ANN as per McGhee (2018) – see previous slide
3. Calculate local volatility surface using Dupire $\sigma_{LV}(t, K)$
4. Apply mixing weight ω : $\xi \to \omega\xi$ (some quants also $\rho \to \omega\rho$)
5. Evolve the density (Fokker-Planck equation) on a 2 factor Finite Difference to determine the scaling:

$$\lambda(t, K) = \frac{\sigma_{LV}(t, K)}{\sqrt{\mathbb{E}[\sigma^2(t)|S_t = K]}}$$

6. Price contract by solving a 2 factor Finite Difference with scaled stochastic volatility process

A few key features used in the Neural Network in McGhee (2018)

Solving a **real problem** (speed up of SABR process) through neural networks:

1. The choice of objective function is to minimize a ”profile” of implied vols (the vol smile)
2. Choice is smooth activation function as soft-plus
3. The choice of domain space to ensure relevance of domain to valuation of the function.
4. Old quant tricks to generate risk – 2FPDE risk usually generated from spline interpolation.
5. Generate training data from 2FPDE – quant teams can re-use existing models to generate vast data sets for training.
6. Transfer learning – Train a network to SABR formula, then use weights to training the actual SABR process to 2FPDE data.

ANN SABR as in McGhee (2018)

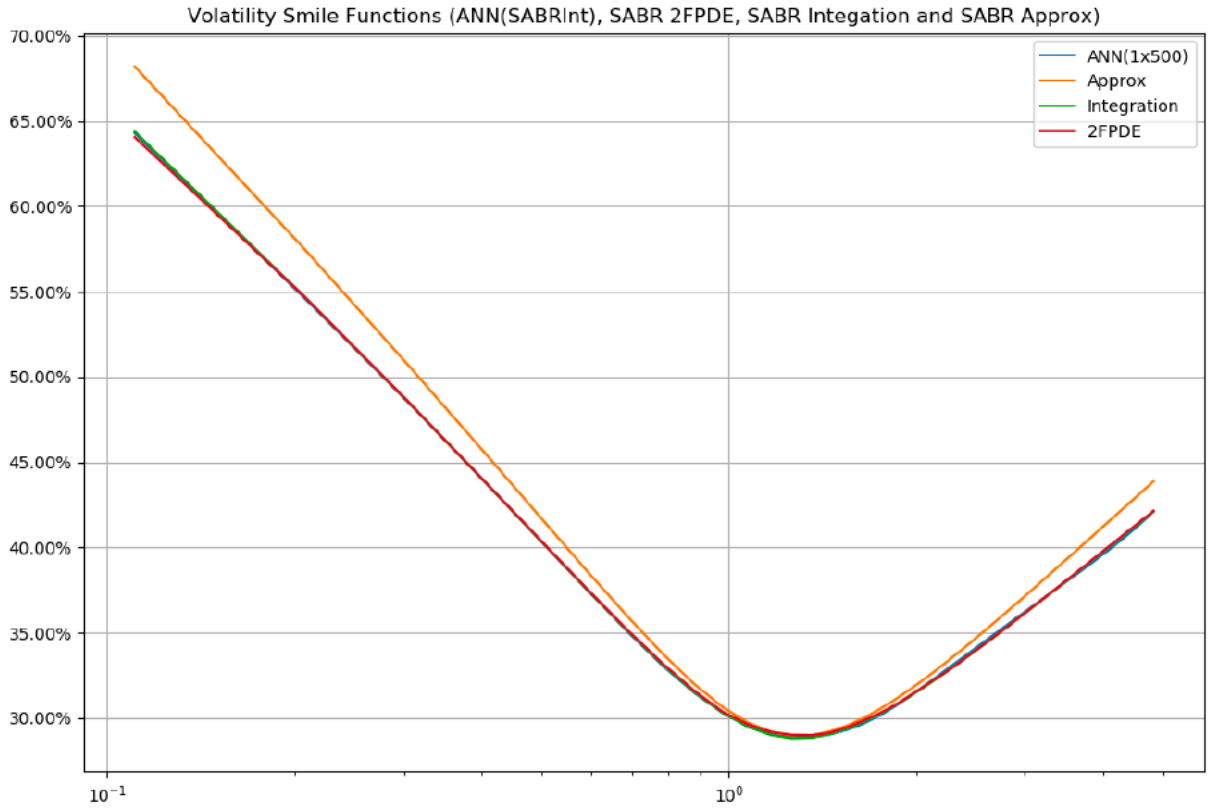


Figure 7: Comparing the SABR Approximation, SABR Integration, SABR 2FPDE and an ANN trained on the SABR Integration scheme for $T = 540d$, $\sigma(0) = 30\%$, $\xi = 60\%$ and $\rho = -35\%$.

High accuracy for Smile, CDF, PDF

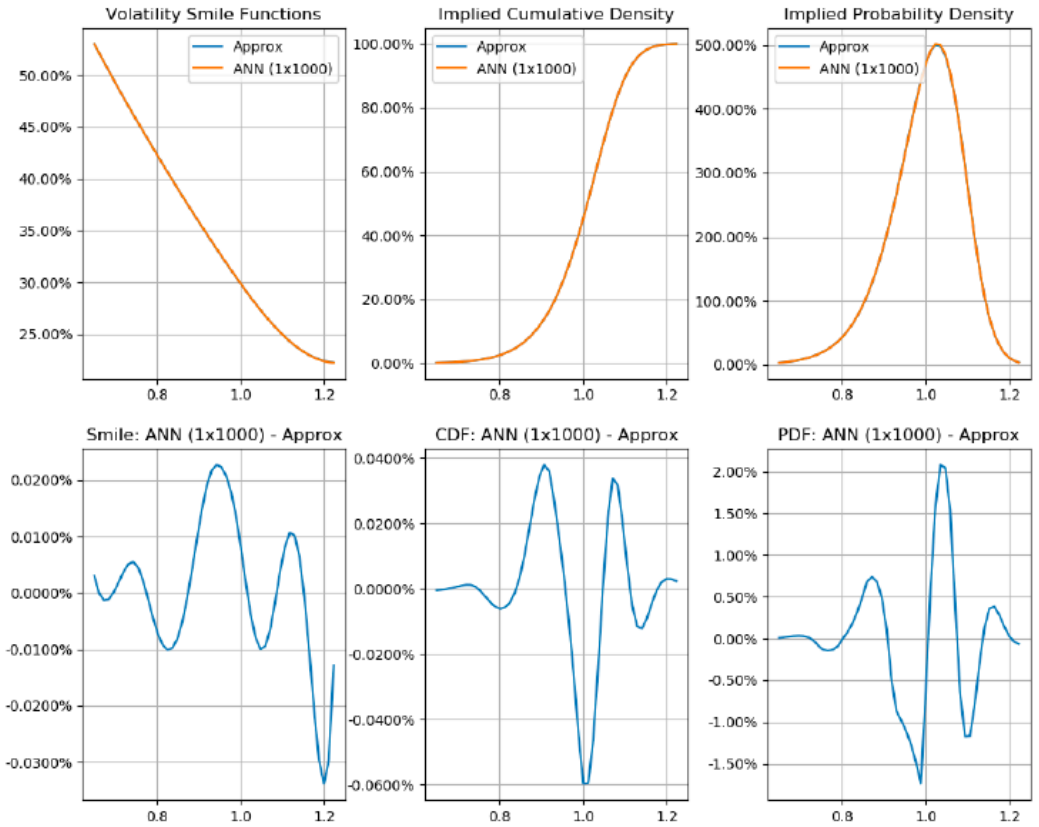


Figure 5: Smiles, CDFs and PDFs ($T = 1M$, $\sigma_0 = 30\%$, $\xi = 150\%$, $\rho = -75\%$)

The objective of our approach for LSVol

- To significantly speed up calculations of price and risk for Exotic Options under LSVol.
- By training a (deep) neural network which can represent an Exotic Option contract accurately (for all features of that contract).
- Choice of domain is key!!!
  - The smaller the domain, the easier the problem… but we need to think in a production environment and consider all scenarios.
- Once trained, the DNN can be used in lieu of pricing/risk.

One Touch as example of path-dependent Exotic Option

Example: One Touch option in foreign exchange (FX):

- Assume the foreign exchange rate is S(0) = 1.300 USD per 1 GBP (1.300 GBPUSD).
- Payout of $1M if the GBPUSD rate touches level B = 1.400 by 31st of October!!!

$$\Pi(S_t, T, B) = 1_{\{\max S_t \geq B : t \leq T\}}$$

Generating the training data

- Challenge: to price One Touch contracts using deep neural network by training prices generated by an LSVol model produced by an ADI Finite Difference scheme
- Function to train depends on 6 scalar inputs and 1 vector input

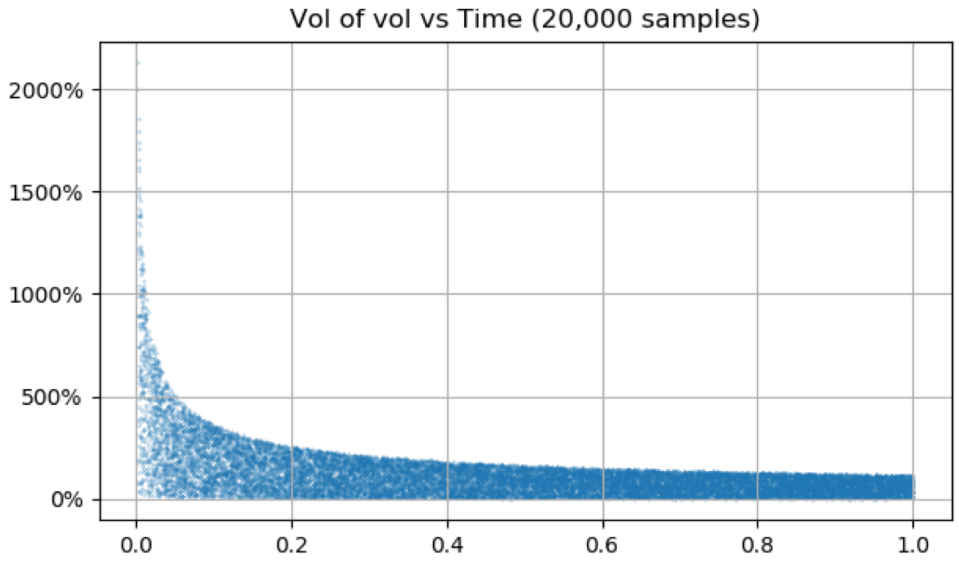
$$\{PV_1, PV_2, \dots, PV_{10}\} = F(\sigma_0, \xi, \rho, \omega, \mu, T, \{B_1, B_2, \dots, B_{10}\})$$

- Generate 200,000 training "profiles" and 50,000 test "profiles" using the known solution from the Finite Difference Scheme
- Use of profiles increases the rate of learning as the network is provided with structural features of the function
- The input parameter space covers a wide range of FX market data as observed in past 20-40 years

Choosing the input domain space

- Time: [1day , 365 days]
- Mu (drift): [-10%, 10%]
- Initial vol: [1%, 50%]
- Vol-of-Vol at 1 month: [5%, 400%], scaled for other tenors, see *McGhee (2018)*
- Rho (correlation): [-95%, 95%]
- Mixing weight: [0%, 100%]
- Barrier/Spot ratio is chosen to cover the range of prices for the given parameter set

Vol-of-Vol and other parameters domain space (McGhee)



$$\xi_t = \xi_s \times \sqrt{\frac{T_s}{T_t}},$$

Tenor	1W	1M	3M	6M	1Y
Maximum	833%	400%	231%	163%	115%
Minimum	10%	5%	3%	2%	1%

$$\sigma(T, \eta_\sigma) = \sigma(0) \exp\left(-\frac{1}{2}\xi^2 T + \xi \eta_\sigma \sqrt{T}\right)$$

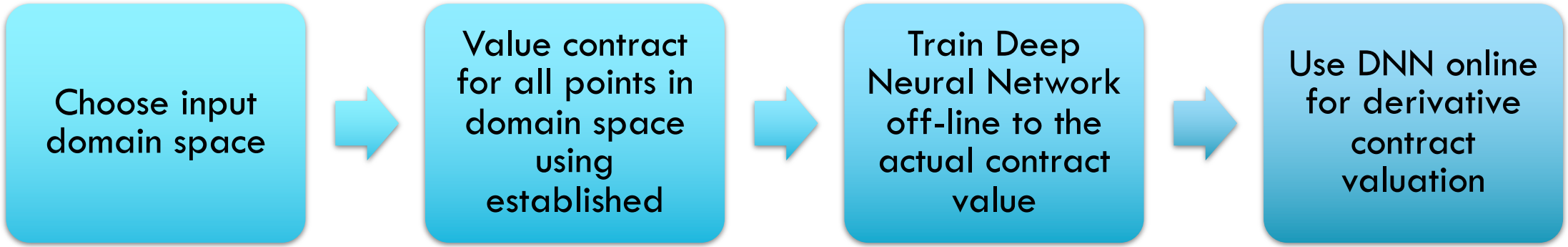
T	2W	1M	2M	3M	6M	9M	1Y
ξ	147.2%	100%	70.7%	57.7%	40.8%	33.3%	28.9%
$K(\eta_S = +3.5, \eta_\sigma = 1.5)$	1.1128	1.1697	1.2468	1.3090	1.4589	1.5837	1.6956
$K(\eta_S = -3.5, \eta_\sigma = 1.5)$	0.8407	0.7741	0.6956	0.6404	0.5309	0.4591	0.4059

Table 3: Strike range for $\sigma_0 = 20\%$ and $\rho = -50\%$

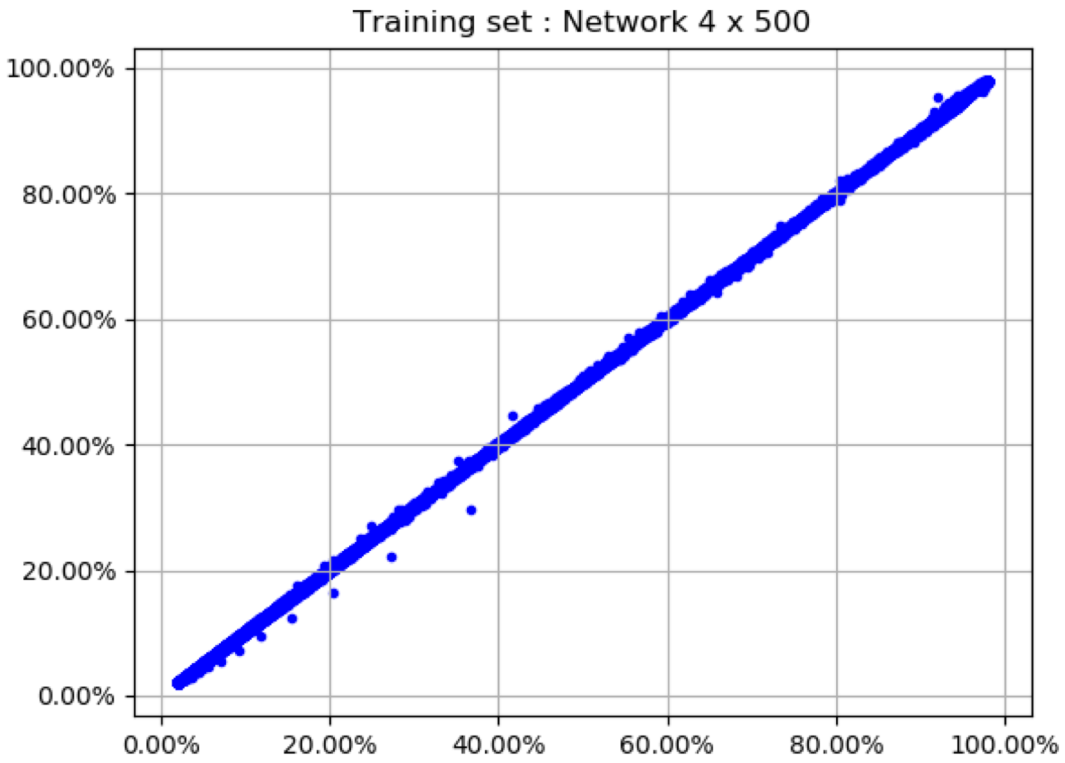
Deep Neural Network architecture

- We have investigated the effectiveness of networks with increasing depth (1 - 5 hidden layers of 500 nodes)
- The activation function is "softplus" $f(x) = \ln(e^x + 1)$ - same as McGhee (2018).
- The networks were trained in Python using Tensor Flow via Keras
- The optimiser is Adam and the loss function is the mean squared error
- The learning rate input varies according to an algorithm to increase speed of convergence
- Each network can take several days to train using an NVIDIA GE Force GTX 1060 GPU

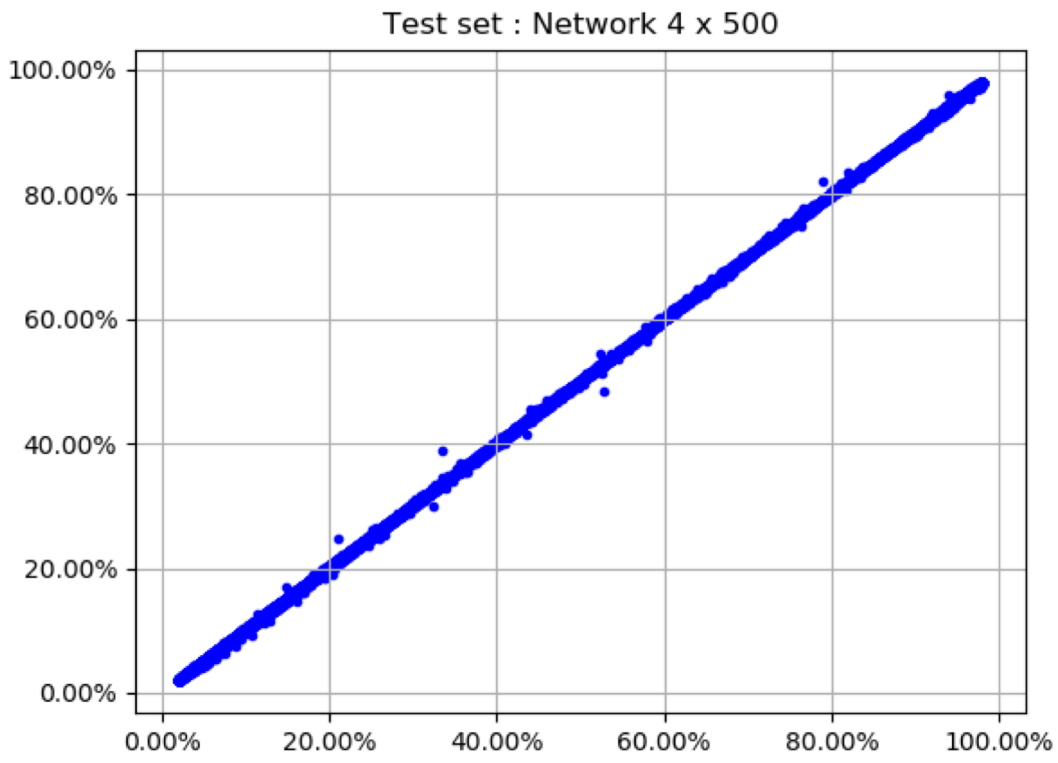
The method


Choose input domain space
Value contract for all points in domain space using established
Train Deep Neural Network off-line to the actual contract value
Use DNN online for derivative contract valuation

Training and Test set errors



One-Touch prices DNN predict x actual (PDE)



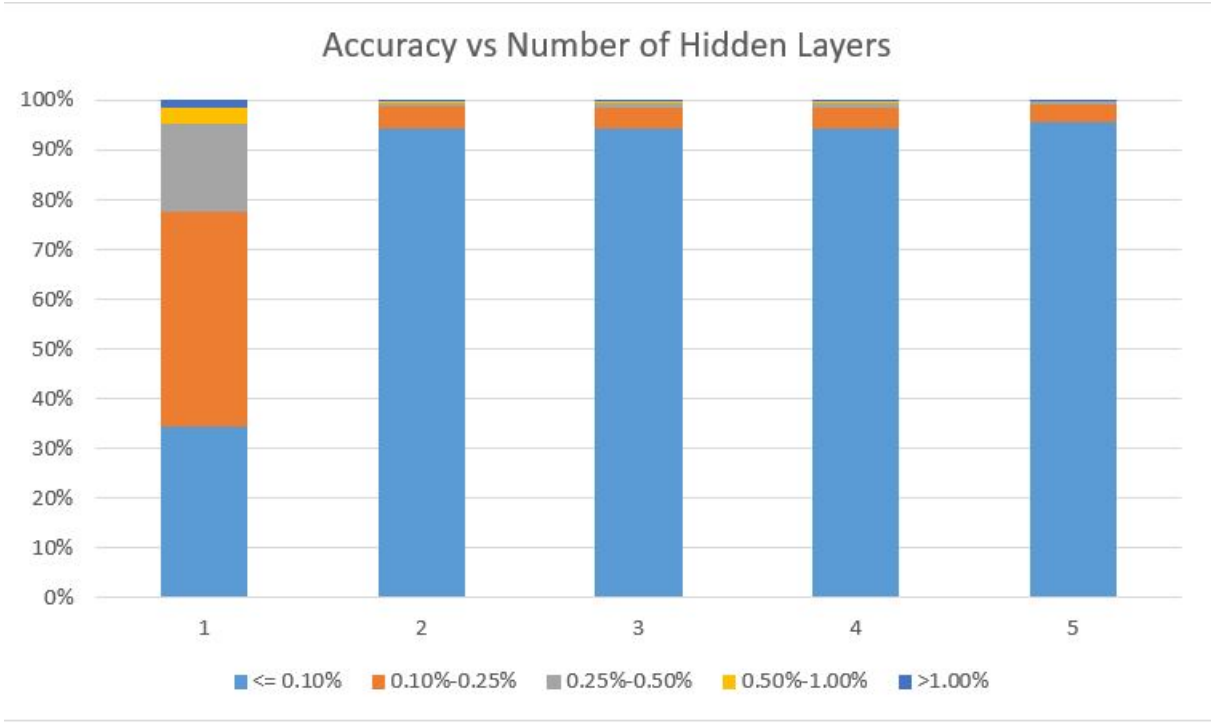
One-Touch prices DNN predict x actual (PDE)

Accuracy for different layers

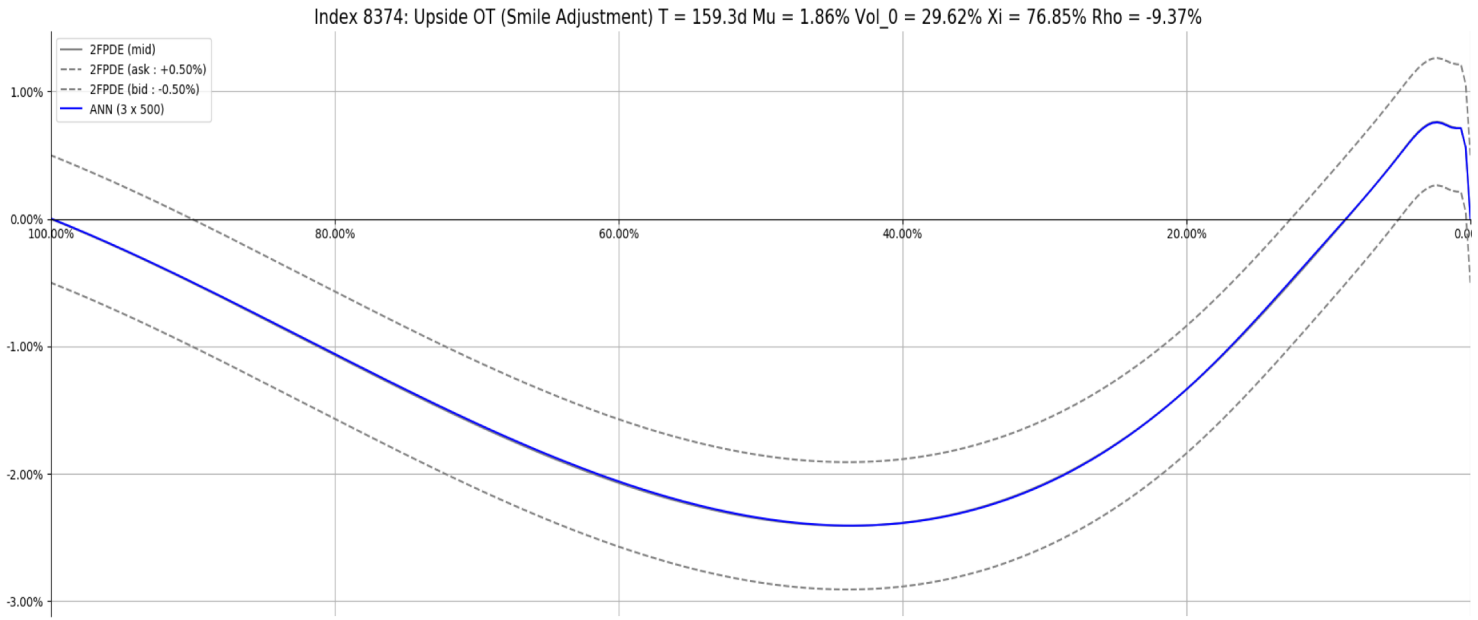
	1x500	2x500	3x500	4x500	5x500
<= 0.10%	34.54%	94.34%	94.29%	94.43%	95.53%
0.10%-0.25%	43.16%	4.36%	4.32%	4.24%	3.51%
0.25%-0.50%	17.73%	0.85%	0.89%	0.86%	0.63%
0.50%-1.00%	3.08%	0.28%	0.31%	0.31%	0.20%
>1.00%	1.49%	0.17%	0.18%	0.16%	0.13%
% within bid-ask	95.43%	99.55%	99.51%	99.53%	99.67%

Maximum absolute error on a per profile basis

Accuracy for different layers

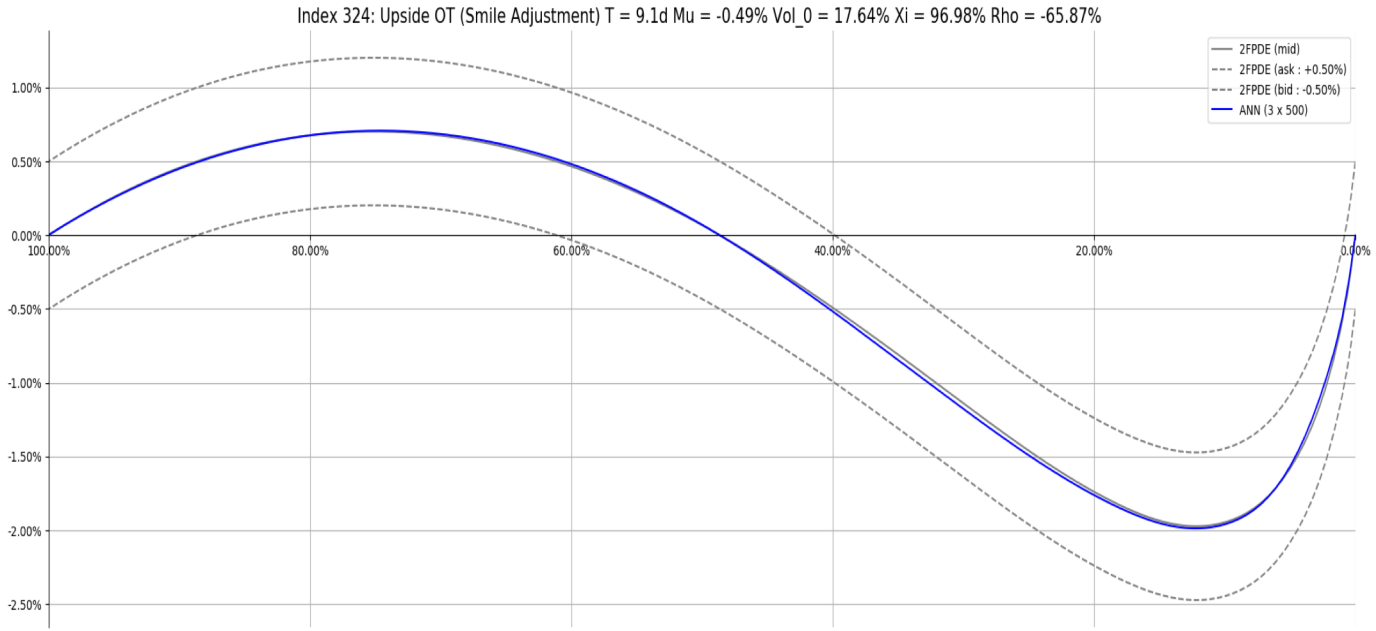


Accuracy Part I: example of convergence of the DNN model



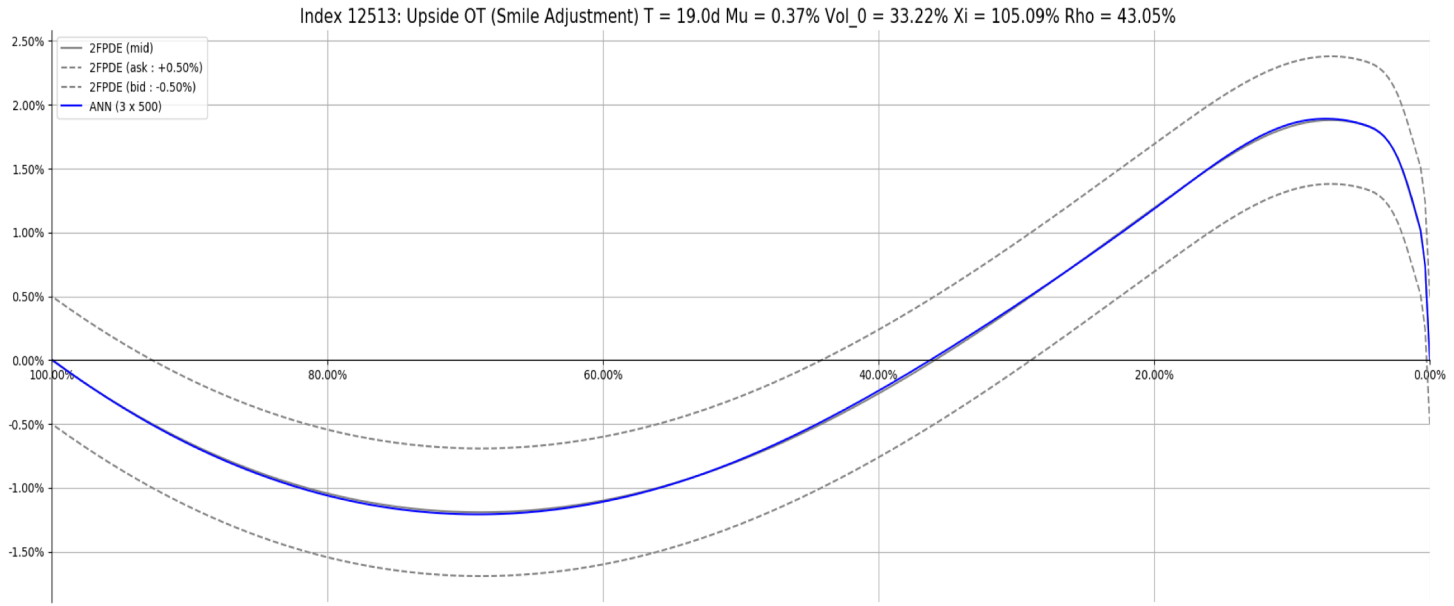
One Touch Smile Adjustment to Black-Scholes

Accuracy Part II: example of convergence of the DNN model



One Touch Smile Adjustment to Black-Scholes

Accuracy Part III: example of convergence of the DNN model



One Touch Smile Adjustment to Black-Scholes

Extension beyond constant parameters

- In practice, there is a term structure of smile parameters which are used within exotic pricing
- The results shown so far, assume constant parameters
- However, volatility surface parameters are not independent across tenors
- 10 tenors of 3 smile parameters does not correspond to 30 independent parameters
- Volatility surfaces exhibit dominant driving factors which significantly reduce the dimension of the problem whilst introducing realistic term structures
- The efficacy of this approach will be explored at a future presentation

Generation of risk

- The procedure of offline training means you don't need to necessarily produce risk from the same model you produce valuation from
- There is no requirement for the DNN to have stable risk – although see Savine (2019)
- Risk can equally well be trained by a different network e.g. DNN(OT, Vega), DNN(OT, Vanna), etc
- Other techniques e.g. spline functions on DNN profiles, see McGhee (2018), have been shown to product accurate first and second order risk

Conclusions

- There is an increasing use of deep neural networks as universal approximator of complex functions in the derivatives pricing space
- Paradigm shift : off-line approximation of complex multivariate functions
- This is the first example of exploring the application of deep learning models for accurate pricing of *path-dependent* Exotic Options using standard models (*LSVol*) beyond Black-Scholes for a realistic range of parameters.
- Once trained off-line, the speed of a risk calculation can be of the order of 20,000 x faster than currently used models across the trading floor (from days to 10mins…)
- Deep learning models also open the possibility to explore even more sophisticated representation of market dynamics by removing barriers on existing performance bottle-necks

THANK YOU

Katia.Babbar@maths.ox.ac.uk