Pricing options on flow forwards by neural networks in Hilbert space

Nils Detering
University of California, Santa Barbara

together with Fred Espen Benth (UiO), and Luca Galimberti (NTNU)

BFS One World Seminar
November 24th, 2022

Contents

Introduction Pricing Problem

- In electricity markets (NYMEX, CME, EEX and NordPool) one trades options on forward contracts delivering over a period of time $[T_1, T_2]$
- For option pricing: model forward prices across different delivery periods to avoid calendar arbitrage
- Denote the price of the forward at time $t \leq T$ by $\hat{F}(t, T_1, T_2)$
- Trick: turn to artificial fixed delivery contracts $F(t, T)$ and retrieve

$$\hat{F}(t, T_1, T_2) = \frac{1}{T_2 - T_1} \int_{T_1}^{T_2} F(t, T) dT$$

- Turn to Musiela parametrization $X(t, \xi) = F(t, t + \xi)$ and model forward curve as a stochastic process $\{X(t, \cdot)\}_{t \geq 0}$.

Introduction Pricing Problem

We need a state space $\mathfrak{X}$:

- Space of weakly differentiable functions $x : \mathbb{R}_+ \to \mathbb{R}$ s.t.

$$|x|_w^2 = x(0)^2 + \int_0^\infty w(\xi) x'(\xi)^2 d\xi < \infty$$

  where the weight function w is non-decreasing with $w(0) = 1$.

We need a dynamics:

- Assume that log forward Y satisfies the SPDE

$$dY(t) = \partial_\xi Y(t) dt + \mu(t, Y(t)) dt + \sigma(t, Y(t)) dW(t)$$

- With "$\mathcal{S}_t = \exp(t\partial_\xi)$" mild solution

$$Y(t) = \mathcal{S}_t y + \int_0^t \mathcal{S}_{t-s} \mu(s, Y(s)) ds + \int_0^t \mathcal{S}_{t-s} \sigma(s, Y(s)) dW(s) \quad (1)$$

Introduction Pricing Problem

- And we let the forward curve be

$$X(t) := \exp(Y(t)).$$

Turns out that $\mathfrak{X}$ is a Banach algebra, so that X is a process in $\mathfrak{X}$ as well. We have that

$$F(t,T) = \delta_{T-t} X(t) = \exp(Y(t, T-t)).$$

- Option price for european option with maturity τ

$$V(t,x) = \mathbb{E}[q(X^{t,x}(\tau))]$$

where $q : \mathfrak{X} \to \mathbb{R}$ is Lipschitz continuous function and $X^{t,x}$ means $X^{t,x}(t) = x$, x current forward curve.

Introduction Pricing Problem

How to calculate $V(t,\cdot)$?

- Introduce a measure μ on $\mathfrak{X}$ and assume

$$\mathbb{E}\left[\int_{\mathfrak{X}} q(X^{t,x}(\tau))^2 \mu(dx)\right] < \infty.$$

- For $g \in L^2(\mu)$ it follows

$$\begin{aligned} & \mathbb{E}\left[\int_{\mathfrak{X}} |q(X^{t,x}(\tau)) - g(x)|^2 \mu(dx)\right] \\ = & \int_{\mathfrak{X}} \mathrm{Var}(q(X^{t,x}(\tau)))\mu(dx) + \int_{\mathfrak{X}} |V(t,x) - g(x)|^2 \mu(dx) \end{aligned}$$

- Thus

$$V(t,\cdot) = \arg \min_{g \in L^2(\mu)} \mathbb{E}\left[\int_{\mathfrak{X}} |q(X^{t,x}(\tau)) - g(x)|^2 \mu(dx)\right].$$

Introduction Pricing Problem

- We then show that

$$I : L^2(\mu) \to \mathbb{R}, \quad g \mapsto \mathbb{E}\left[\int_{\mathfrak{X}} |q(X^{t,x}(\tau)) - g(x)|^2 \mu(dx)\right]$$

  is locally Lipschitz, in particular continuous on $L^2_{lip}(\mu)$,

- and that

$$\min_{L^2(\mu)} I = \min_{L^2_{lip}(\mu)} I = \inf_{L^2_{lip}(\mu)} I$$

  for μ with compact support.

- Plan: Instead of minimizing over L^2_{lip}, we choose a smaller set $\mathfrak{N}(\sigma)$ of suitable neural networks, and show that

$$\inf_{L^2_{lip}} I = \inf_{\mathfrak{N}(\sigma)} I.$$

- Then find minimizing neural network by stochastic gradient descent based on samples $X_i^{t,x_i}(\tau)$.

Talk based on

1. Papers:
   - *Neural Networks in Fréchet spaces* (with F.E. Benth und L. Galimberti), accepted Annals of Mathematics and Artificial Intelligence, arXiv:2109.13512 [math.FA]
   - *Pricing options on flow forwards by neural networks in Hilbert space* (with F.E. Benth und L. Galimberti), arXiv:2202.11606 [q-fin.PR]
2. Code:
   - https://github.com/ncdetering/FlowForwardsNumerics

Contents

What is a neural network?

Standard neural network (but there are many variations)

- Fixed continuous nonlinear function $\sigma : \mathbb{R} \to \mathbb{R}$,
- a vector $a \in \mathbb{R}^n$,
- scalar $l, b \in \mathbb{R}$.

A *neuron* is a function $\mathcal{N}_{\ell,a,b} \in C(\mathbb{R}^n; \mathbb{R})$ defined by

$$x \mapsto \ell\sigma(a^\top x + b).$$

A *one layer neural network* (NN) is then a finite sum of neurons

$$\sum_{i=1}^{N} \mathcal{N}_{\ell_i,a_i,b_i}(x) = \sum_{i=1}^{N} \ell_i\sigma(a_i^\top x + b_i).$$

Universal Approximation Theorem

- The universal approximation theorem states conditions on the activation function σ which ensure that the linear space of functions generated by the neurons

$$\mathfrak{N}(\sigma) := \text{span}\{\mathcal{N}_{\ell,a,b}; \ell, b \in \mathbb{R}, a \in \mathbb{R}^n\}$$

  is dense with respect to the topology of uniform convergence on compacts.

- This means that for every $f \in C(\mathbb{R}^n; \mathbb{R})$ and compact subset $K \subset \mathbb{R}^n$ and a given $\varepsilon > 0$, there exists $N \in \mathbb{N}$ and $\ell_i, b_i \in \mathbb{R}, a_i \in \mathbb{R}^n$ for $i = 1, \ldots, N$ such that

$$\sup_{x \in K} |f(x) - \sum_{i=1}^{N} \mathcal{N}_{\ell_i, a_i, b_i}(x)| \leq \varepsilon.$$

Universal Approximation Theorem

Possibly the most widely known property of σ that was shown in

- Cybenko (1989)
- Hornik et al. (1989)

to lead to the density of $\mathfrak{N}(\sigma) \subset C(\mathbb{R}^n; \mathbb{R})$ is the *sigmoid* property, which requires σ to be such that

$$\lim_{t \to \infty} \sigma(t) = 1, \qquad \lim_{t \to -\infty} \sigma(t) = 0$$

This condition has later been relaxed to

- a boundedness condition: Funahashi (1989)
- and a non-polynomial condition: Leshno et al. (1993)
- recently, Kratsios (2021) for unified approach (wide class of network architectures)

The paper Berner et al. (2021) provides good overview of mathematical results related to deep learning.

A Fréchet neuron

Towards approximation of functions $f \in C(\mathfrak{X}; \mathbb{R})$, in the definition of a neuron, we replace

- $a^\top x + b$ by an affine function on $\mathfrak{X}$, i.e. $Ax + b, A \in \mathcal{L}(\mathfrak{X}), b \in \mathfrak{X}$.
- the activation function $\sigma : \mathbb{R} \to \mathbb{R}$ by a function in $C(\mathfrak{X}; \mathfrak{X})$,
- the scalar ℓ by a linear form, i.e. $\ell : \mathfrak{X} \to \mathbb{R}$.

We then define a *neuron* $\mathcal{N}_{\ell,A,b}$ by

$$\mathcal{N}_{\ell,A,b}(x) = \langle \ell, \sigma(Ax + b) \rangle,$$

$\langle \cdot, \cdot \rangle$ canonical pairing between $\mathfrak{X}'$ and $\mathfrak{X}$ ($\mathfrak{X}'$ denoting the topological dual of $\mathfrak{X}$).

Overview results I

Recall that any $\psi \in \mathfrak{X}' \setminus \{0\}$ defines hyperplane in $\mathfrak{X}$ by

$$\Psi_0 = \ker(\psi),$$

which sits in $\mathfrak{X}$ between the two half-spaces

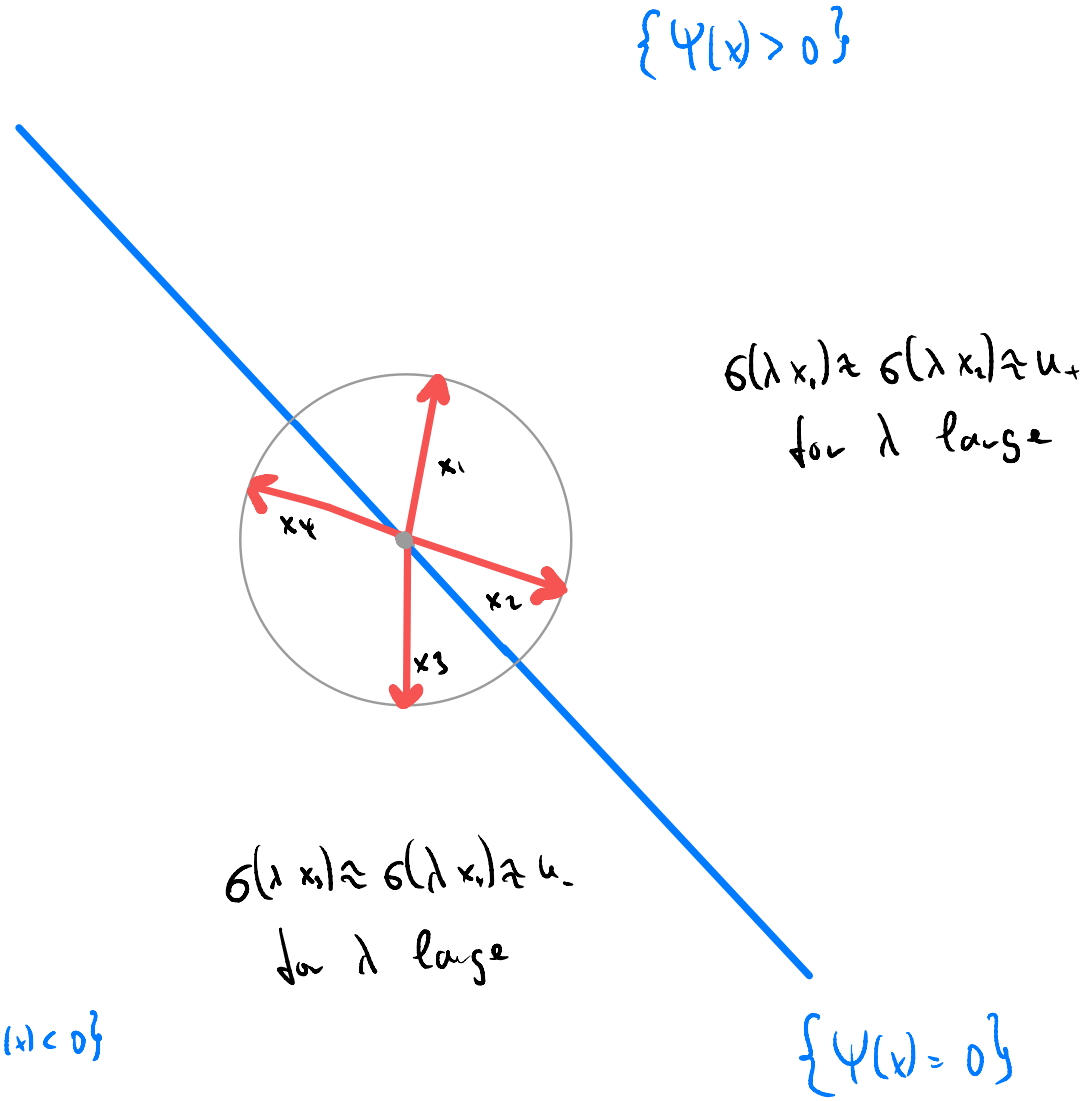
$$\Psi_+ = \{x \in \mathfrak{X}; \langle \psi, x \rangle > 0\} \text{ and } \Psi_- = \{x \in \mathfrak{X}; \langle \psi, x \rangle < 0\}$$

Definition (Separating property)

We call $\sigma : \mathfrak{X} \to \mathfrak{X}$ separating if there exist $\psi \in \mathfrak{X}' \setminus \{0\}$ and $u_+, u_-, u_0 \in \mathfrak{X}$ such that either $u_+ \notin \text{span}\{u_0, u_-\}$ or $u_- \notin \text{span}\{u_0, u_+\}$ and

$$\begin{cases} \lim_{\lambda \to \infty} \sigma(\lambda x) = u_+, \text{ if } x \in \Psi_+ \\ \lim_{\lambda \to \infty} \sigma(\lambda x) = u_-, \text{ if } x \in \Psi_- \\ \lim_{\lambda \to \infty} \sigma(\lambda x) = u_0, \text{ if } x \in \Psi_0. \end{cases}$$

Overview results I



Example Separating σ

Consider a function $\beta \in \mathit{Lip}(\mathbb{R};\mathbb{R})$ such that

$$\lim_{\xi\to\infty} \beta(\xi) = 1, \quad \lim_{\xi\to-\infty} \beta(\xi) = 0, \ \beta(0) = 0.$$

Let $\psi \in \mathfrak{X}' \setminus \{0\}$ and $z \in \mathfrak{X}, z \neq 0$. Define

$$\sigma(x) = \beta(\psi(x))z, \quad x \in \mathfrak{X}.$$

Evidently, σ is continuous, can be shown von Neumann-bounded and Separating.

Overview results II

Theorem (Benth-D-Galimberti, 2021)

Let $\sigma : \mathfrak{X} \to \mathfrak{X}$ continuous, separating, and von Neumann bounded. The subspace

$$\mathfrak{N}(\sigma) := \text{span}\{\mathcal{N}_{\ell,A,b}; \ell \in \mathfrak{X}', A \in \mathcal{L}(\mathfrak{X}), b \in \mathfrak{X}\}$$

is dense in $C(\mathfrak{X}; \mathbb{R})$ under the topology for uniform convergence on compacts.

Similar results hold for:

- Allow $f \in C(\mathfrak{X}; \mathfrak{Y})$, where $\mathfrak{Y}$ is Banach space,
- n-layer neural nets based on

$$\langle \ell, (\sigma \circ T_1 \circ \cdots \circ \sigma \circ T_n)(x)$$

with $T_j(x) := A_j x + b_j$.

Overview results III

Theorem (Benth-D-Galimberti, 2021)

Let $\mathfrak{X}$ Fréchet space with Schauder basis $(e_k)_{k\in\mathbb{N}}$, i.e. every $x \in \mathfrak{X}$ can be written as $\sum_{k=1}^{\infty} x_k e_k$. Let further σ fulfill special Lipchitz condition. Let $f \in C(\mathfrak{X};\mathbb{R})$, $K \subset \mathfrak{X}$ compact and $\varepsilon > 0$. Assume

$$\mathcal{N}^{\varepsilon}(x) = \sum_{j=1}^{M} \langle \ell_j, \sigma(A_j x + b_j)\rangle, \quad x \in \mathfrak{X}$$

with $\ell_j \in \mathfrak{X}'$, $A_j \in \mathcal{L}(\mathfrak{X})$ and $b_j \in \mathfrak{X}$ such that

$$\sup_{x\in K} |f(x) - \mathcal{N}^{\varepsilon}(x)| < \varepsilon.$$

Fix $\delta > 0$. Then there exists $N_* = N_*(\mathcal{N}^{\epsilon}, \delta) \in \mathbb{N}$ s.t. for $N \geq N_*$

$$\sup_{x\in K} |f(x) - \sum_{j=1}^{M} \langle \ell_j \circ \Pi_N, \sigma(\Pi_N A_j \Pi_N x + \Pi_N b_j)\rangle| < \varepsilon + \delta.$$

Related approaches infinite dimensional neural network

- Sandberg (1991) non-linear functionals on a space of functions from $\mathbb{N} \cup \{0\}$ to $\mathbb{N}$.
- Chen and Chen (1993, 1995) consider the approximation of non-linear operators $f : K \subset \mathfrak{X} \to \mathbb{R}$, where $\mathfrak{X}$ is Banach, K is compact and f is continuous.
- (DeepONets) Lu et al. (2021), and Lanthaler et al. (2022) approximation of operators between Banach spaces of continuous functions on compact subsets of $\mathbb{R}^n$.
- Kovachki et al. (2021) and Li et al. (2020) propose NN to approximate the solution operator that assigns to a coefficient function for a partial differential equation (PDE) its solution function. Again operators between Banach spaces of functions defined on a bounded domain in $\mathbb{R}^n$.
- Mhaskar and Hahm (1993) derive networks that approximate the functionals on the function spaces $L^p([-1,1]^s)$ for $1 \leq p < \infty$ and $C([-1,1]^s)$ for integer $s \geq 1$.
- Kratsios (2021) considers a space $M(\mathfrak{X}, \mathfrak{Y})$ of functions from a metric space $\mathfrak{X}$ to another metric space $\mathfrak{Y}$ in cases where this functions space is homeomorphic to an infinite-dimensional Fréchet space.

Contents

Pricing options in electricity markets

- Start with

$$\tilde{e}_k(\tau) = \frac{\tau^{(k-2)} \exp(-\tau)}{\Gamma(k-1)},$$

- Use Gram-Schmidt to derive orthonormal sequence $e_1, e_2, \ldots$
- Consider subspace spanned by $\{e_1, e_2, \ldots, e_7\}$
- Choose $K_7 \subset \mathfrak{X}$ by

$$K_7 := \{x \in \mathfrak{X}; x = \sum_{k=1}^{7} x_i e_i, x_i \in [a_i, b_i], a_i < b_i]$$

  equipped with uniform measure.
- Time horizon $\tau = T_1 = 1/12$, delivery end $T_2 = 2/12$
- Call option with $p : \mathfrak{X} \to \mathbb{R}$ given by

$$p(Y(\tau)) = \max\{0, (1/(T_2 - T_1)) \int_0^{T_2-\tau} \exp(Y(\tau)(T))dT - K\}.$$

Implementation: specification of the SPDE

- We consider two specifications for the coefficients and for the Wiener process in the dynamics of Y:
  - 1-dimensional noise
  - 7-dimensional noise

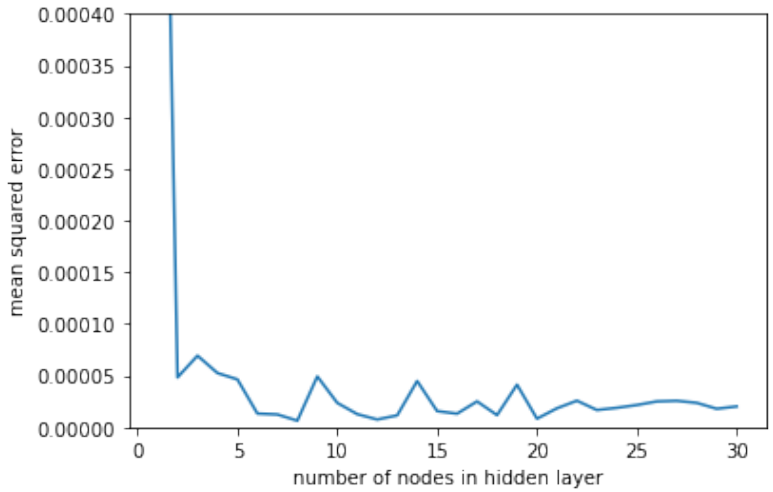
Training the network

- Let
$$x_s \in [-1/2, 1/2]^7 \simeq K_7, \quad s = 1, \ldots, S = 10{,}000{,}000$$
where the x_s are generated independently and uniformly distributed (i.e. $\sim \mu$)
- We derive $q(X_s^{t,x_s}(\tau))$ under both models described above. Namely,
$$\mathfrak{X} \supset K_7 \ni x_s \mapsto X_s^{t,x_s}(\tau) \in \mathfrak{X} \mapsto q(X_s^{t,x_s}(\tau)) \in \mathbb{R}$$
- The training set then consists of the input-output pairs
$$(x_s, q(X_s^{t,x_s}(\tau))) \in [-1/2, 1/2]^7 \times \mathbb{R}, \quad s = 1, \ldots, S$$
- We use this training set to fit several neural networks with the number of hidden nodes ranging from 1 to 30
- Batch size: 10,000, number of epochs: 50,
- $\sigma(x) = \beta(\psi(x))z$ with $\beta(y) = \max\{0, 1 - \exp(-y)\}$.

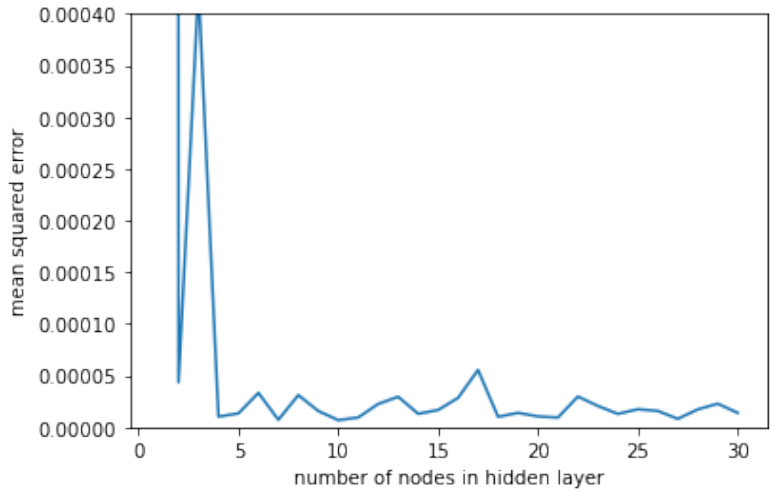
Implementation: the test set

- We then generate a test set of size 10,000.
- In contrast to the training set, the test set is composed of pairs consisting of starting values x_s^{test} and option prices $\mathbb{E}[q(X^{t,x_s^{\text{test}}}(\tau))]$ for $s = 1, \ldots, 10{,}000$.
- The calculation of the benchmark prices $\mathbb{E}[q(X^{t,x_s^{\text{test}}}(\tau))]$ is based on a Monte Carlo simulation with 100,000 simulations each.
- We finally calculate for each of the fitted models the mean squared error with respect to the test set.

Results: mean squared error



(a)



(b)

Figure: Mean squared error for our network architecture with number of neurons ranging from 1 to 30 (a) One dimensional noise specification. Average error over the networks with the number of nodes ranging from 10 to 30 is $2.047 \cdot 10^{-5}$. (b) Multi dimensional noise specification. Average error over the networks with the number of nodes ranging from 10 to 30 $1.859 \cdot 10^{-5}$. .

Results: varying coefficient x_1

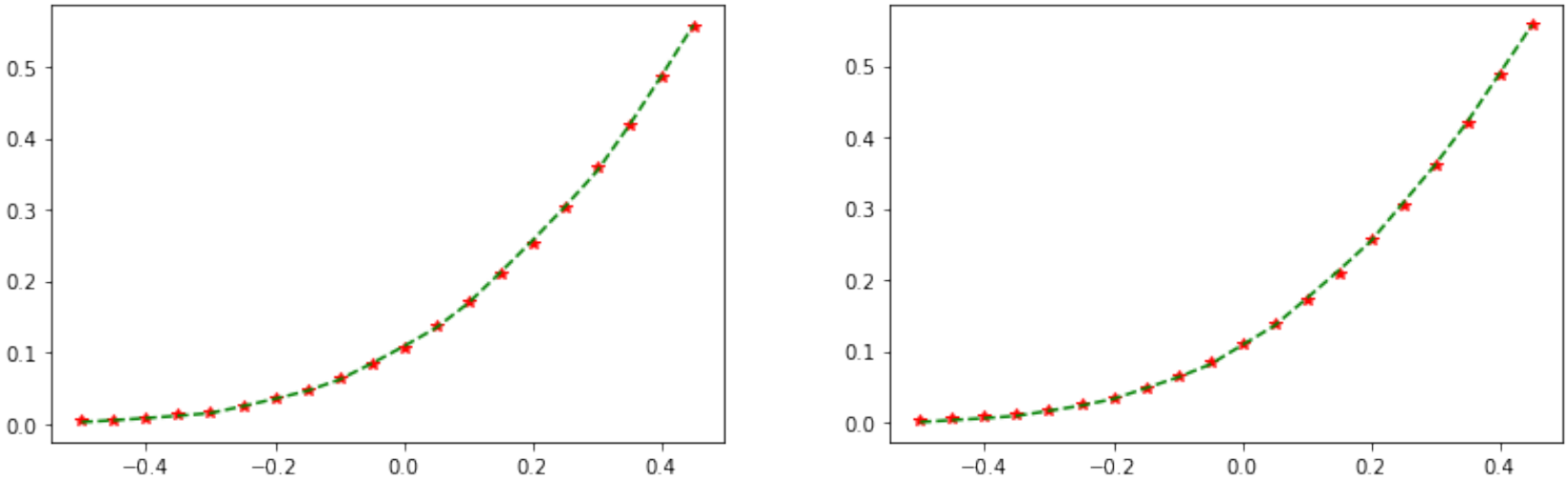


Figure: We plot the option price for initial log forward curves with varying basis coefficient x_1 and $x_2 = \cdots = x_7 = 0$. The green line describes the option price as determined by the neural network while the red stars are option prices as determined by Monte Carlo simulations. (a) One dimensional noise. (b) Multi dimensional noise.

Results: comparison with classical neural network

- We compare the accuracy of our approach to the one obtained with a classical neural network with one dimensional activation and where instead of using the basis coefficients as inputs to the network we discretize the initial function x on an equally spaced grid.
- Recall that each sample x_s represents $x_s(\xi) = \sum_{k=1}^{7} x_{s,k} e_k(\xi)$.
- We evaluate the function on a grid of equally spaced points

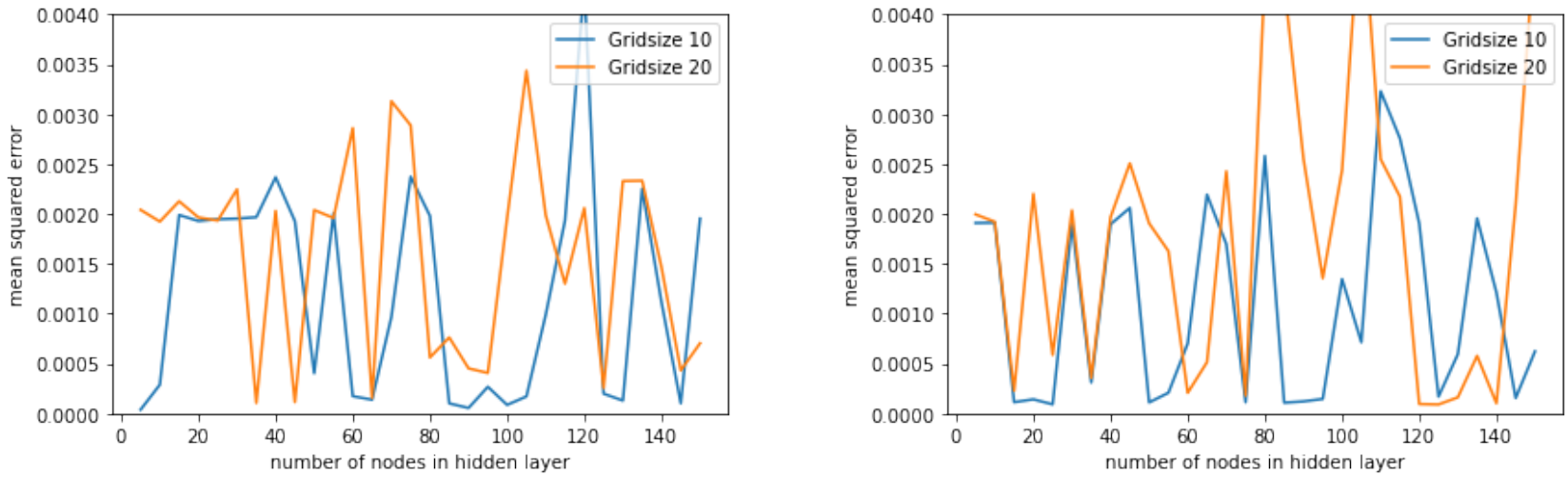
$$0 = \xi_0 < \cdots < \xi_D, \quad D \in \{10, 20\}$$

and analyze the accuracy with a standard one layer neural network with input dimension D trained with the input-output pairs $((x_s(\xi_0), \ldots, x_s(\xi_D)), q(X_s^{t,x_s}(\tau)))$.

Results: comparison with classical neural network

- We fit neural networks with the number of hidden nodes ranging from 5 to 150.
- For $D = 10$ the number of parameters to be fitted for these networks ranges from 60 (5 nodes) to 1,800 (150 nodes)
- This is comparable to the previous setting where the number of parameters ranged from 63 (1 node) to 1,890 (30 nodes).
- For $D = 20$ the parameters range from 110 (5 nodes) to 3,300 (150 nodes).
- We finally display the resulting mean squared errors

Results: comparison with classical neural networks



(a) (b)

Figure: Mean squared error for classical neural network architecture with number of neurons ranging from 5 to 150 (a) One dimensional noise. (b) Multi dimensional noise. As one can see, the mean squared error is significantly larger than with the Hilbert space neural network. Averaging over the networks with at least 20 nodes, the average mean square error is $1.251 \cdot 10^{-3}$ (gridsize 10) and $1.550 \cdot 10^{-3}$ (gridsize 20) for the one dimensional noise, and $1.075 \cdot 10^{-3}$ (gridsize 10) and $1.827 \cdot 10^{-3}$ (gridsize 20) for the multidimensional noise.

Wrapping up

- Our architecture takes into account structure that does not need to be learned by the network itself. It is constructed in terms of interpretable factors (like level, slope and curvature) of the term structure, i.e. $e_1, e_2, \ldots$
- This pays also out in computing sensitivities of the option price V with respect to these factors. (Using a classical NN trained by sampling to compute Greeks is not an easy task.)

References I

Julius Berner, Philipp Grohs, Gitta Kutyniok, and Philipp Petersen. The modern mathematics of deep learning, 2021.

T. Chen and H. Chen. Approximations of continuous functionals by neural networks with application to dynamic systems. *IEEE Transactions on Neural Networks*, 4(6): 910–918, 1993. doi: 10.1109/72.286886.

Tianping Chen and Hong Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4):911–917, 1995. doi: 10.1109/72.392253.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. doi: 10.1007/BF02551274. URL https://doi.org/10.1007/BF02551274.

Ken-Ichi Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192, 1989. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(89)90003-8. URL https://www.sciencedirect.com/science/article/pii/0893608089900038.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(89)90020-8. URL https://www.sciencedirect.com/science/article/pii/0893608089900208.

References II

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces, 2021. URL https://arxiv.org/abs/2108.08481.

Anastasis Kratsios. The universal approximation property. *Annals of Mathematics and Artificial Intelligence*, 89(5):435–469, 2021. doi: 10.1007/s10472-020-09723-1. URL https://doi.org/10.1007/s10472-020-09723-1.

Samuel Lanthaler, Siddhartha Mishra, and George E Karniadakis. Error estimates for DeepONets: a deep learning framework in infinite dimensions. *Transactions of Mathematics and Its Applications*, 6(1), 03 2022. ISSN 2398-4945. doi: 10.1093/imatrm/tnac001. URL https://doi.org/10.1093/imatrm/tnac001. tnac001.

Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993. ISSN 0893-6080. doi: https://doi.org/10.1016/S0893-6080(05)80131-5. URL https://www.sciencedirect.com/science/article/pii/S0893608005801315.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations, 2020. URL https://arxiv.org/abs/2003.03485.

References III

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021. doi: 10.1038/s42256-021-00302-5. URL https://doi.org/10.1038/s42256-021-00302-5.

H. N. Mhaskar and Nahmwoo Hahm. Some new results on neural network approximation. *Neural Networks*, 6(8):1069–1072, 1993.

I.W. Sandberg. Approximation theorems for discrete-time systems. *IEEE Transactions on Circuits and Systems*, 38(5):564–566, 1991. doi: 10.1109/31.76498.

Thank you!

Nils Detering
Department of Statistics and Applied Probability
University of California, Santa Barbara
CA 93106
USA

detering@pstat.ucsb.edu