Neural networks unleashed: Joint SPX/VIX calibration going fast

Giacomo Bormetti
Department of Economics and Management, University of Pavia, Italy
giacomo.bormetti@unipv.it

joint work with

Fabio Baschetti
Department of Economics, University of Verona, Italy
fabio.baschetti@univr.it

Pietro Rossi
Prometeia, Bologna, Italy
pietro.rossi@prometeia.com

Bayes Business School – City University of London
March 5, 2025 - London, UK

Motivation

Traditional one-factor stochastic volatility (SV) models allow a latent process Y_t to implicitly depend on the historical price path of the asset through non-zero correlation with the spot:

$$\frac{\mathrm{d}S(t)}{S(t)} = f(t, Y(t))\mathrm{d}W_1(t)$$

$$Y(t) = Y_0 + \int_0^t \mu(t, u, Y(u))\mathrm{d}u + \int_0^t \nu(t, u, Y(u)) \left(\rho \frac{1}{f(u, Y(u))} \frac{\mathrm{d}S(u)}{S(u)} + \sqrt{1-\rho^2}\mathrm{d}W_u^{\perp} \right)$$

- $\rho \neq \{-1, 0, 1\}$: **partial path-dependency**
- if $f(t, Y(t)) = \sqrt{Y(t)}$ and $\mu(t, u, Y(u))$, $\nu(t, u, Y(u))$ properly specified: **rough Heston model** (El Euch and Rosenbaum (2018)) -> roughness and volatility persistence
- if $f(t, Y(t)) = \sqrt{a + b(Y(t) - c)^2}$ and $\mu(t, u, Y(u))$, $\nu(t, u, Y(u))$ properly specified: **quadratic rough Heston model** (Gatheral *et al.* (2020)) -> Zumbach effect

Volatility is (mostly) path-dependent

Guyon and Lekeufack. *Quantitative Finance*, **23**(9):1221–1258, 2023

$$\frac{\mathrm{d}S(t)}{S(t)} = \mu_t \mathrm{d}t + \sigma(R_{1,t}, R_{2,t})\mathrm{d}W_1(t)$$

$$\sigma(R_{1,t}, R_{2,t}) = \beta_0 + \beta_1 R_{1,t} + \beta_2 \sqrt{R_{2,t}} + \beta_{1,2} R_{1,t}^2 \mathbf{1}_{R_{1,t}>0}$$

$$R_{1,t} = \int_{-\infty}^{t} K_1(t-u)\frac{\mathrm{d}S(u)}{S(u)}$$

$$R_{2,t} = \int_{-\infty}^{t} K_2(t-u)\sigma_u^2 \mathrm{d}u\,.$$

- $\sigma(R_{1,t}, R_{2,t})$ **purely endogenous** and driven by $(R_{1,t}, R_{2,t})$
- lack of probability on the right tail (misspricing of ATM calls):

$$\mathrm{d}S(t)/S(t) = \mu_t \mathrm{d}t + \mathrm{e}^{X_t}\sigma(R_{1,t}, R_{2,t})\mathrm{d}W_1(t)$$

  with X_t Ornstein-Uhlenbeck (OU): **mostly** path-dependent
- $\beta_{1,2}$ very significant when pricing

SPX options, VIX futures, and VIX options

SPX options

$$C_{SPX}(t,T,k) = \mathrm{e}^{-r(T-t)}\mathbb{E}[(S_T - \mathrm{e}^k)^+ | \mathcal{F}_t]\,.$$

The CBOE Volatility Index (VIX)

The VIX is is a popular measure of the market's expected volatility of the SPX index over the next $\Delta = 30$ days. Within a pricing model (with continuous paths)

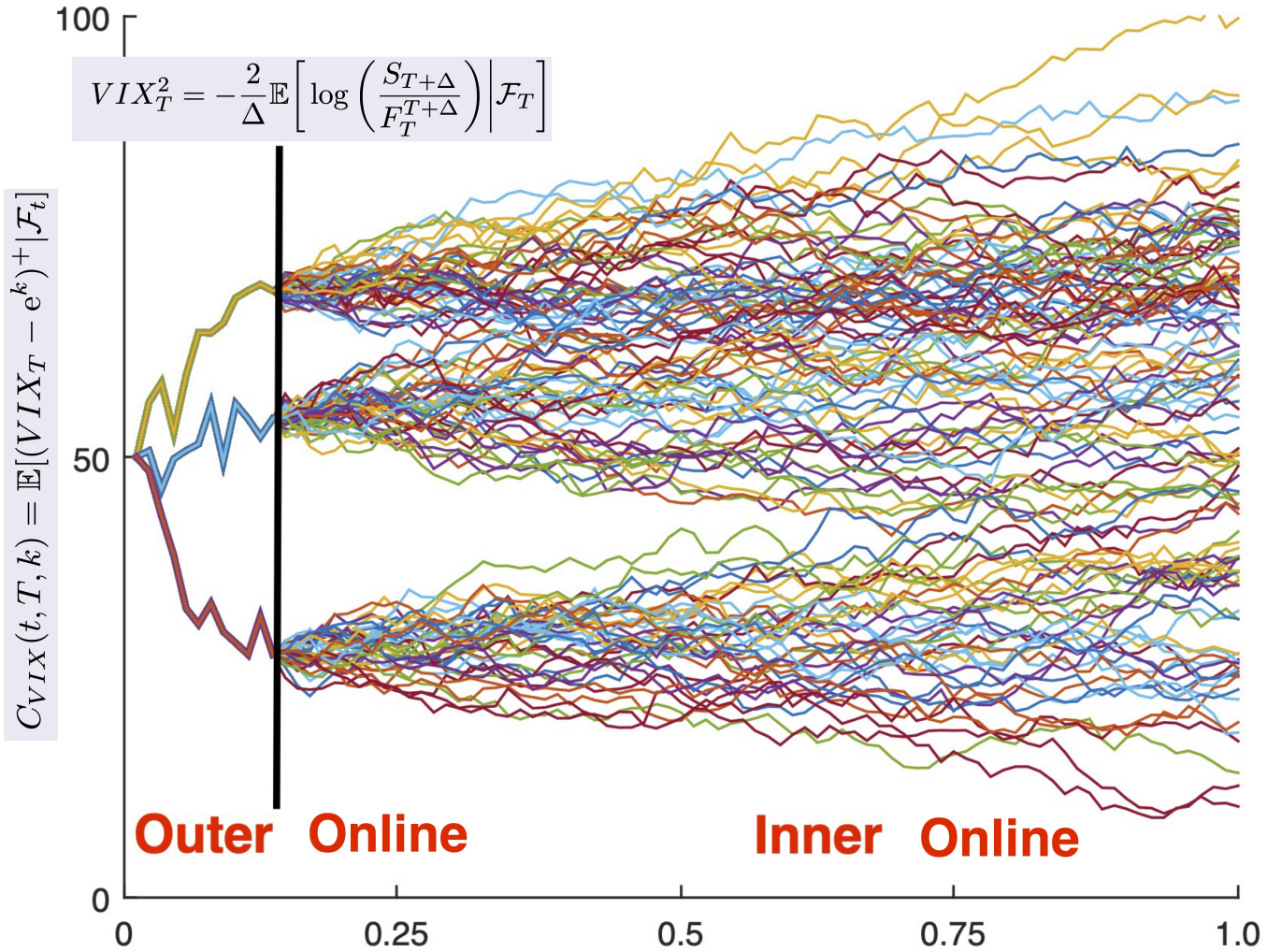
$$VIX_T^2 = -\frac{2}{\Delta}\mathbb{E}\left[\log\left(\frac{S_{T+\Delta}}{F_T^{T+\Delta}}\right)\middle|\mathcal{F}_T\right] = \frac{1}{\Delta}\mathbb{E}\left[\int_T^{T+\Delta}\sigma_t^2 dt\middle|\mathcal{F}_T\right]$$

VIX Derivatives

VIX Futures and Options

$$F_{VIX}(t,T) = \mathbb{E}[VIX_T|\mathcal{F}_t]\,,$$
$$C_{VIX}(t,T,k) = \mathrm{e}^{-r(T-t)}\mathbb{E}[(VIX_T - \mathrm{e}^k)^+|\mathcal{F}_t]\,.$$

Nested Monte Carlo



Outline of the talk

- **I Part**: leveraging the **pointwise Neural Network approach** to pricing
- **II Part**: Joint SPX/VIX calibration of the 4-factor PDV Markov model of Gazzani & Guyon (2024).

I Part

Deep learning with random grids

Two possible routes to calibration

- If the characteristic function (CF) is available in (semi-)closed form:

PROS very efficient machinery to perform computations in Fourier space [FFT-Lewis, SINC from Baschetti et al. *Quant. Finance* (2022)]

CONS the computation of the CF may be slow e.g., rHeston from El Euch and Rosenbaum *Math. Finance* (2019) requires solving a fractional equation. Of course, the rational approximation by Gatheral and Radoičić *IJTAF* (2019) speeds up the procedure.

SOL **one may train a Neural Network to learn the pricing function** to achieve super-fast computation of the implied volatilities and sensitivities

- The model is purely simulative (e.g., rBergomi by Bayer, Priz, Gatheral, *Quant. Finance* (2016) or quadratic rHeston by Gatheral, Jusselin, and Rosenbaum *Risk* (2020), PDV models)

PROS very flexible

CONS Monte Carlo (MC) pricing is computationally too heavy

CONS convergence of the optimization severely affected by MC errors

SOL calibration by feed-forward Neural Networks (Hernandez *Risk* (2017), Horvath, Muguruza, and Tomas *Quant. Finance* (2021))

Calibration by feed-forward Neural Networks

Grid-based methods in the literature (Hernandez (2017), Horvath *et al.* (2021), Rømer *Quant. Finance* (2022))

Train the Neural Network to learn a collection of pixels from the volatility surface over a *fixed* bi-dimensional grid in (T, k)

k/T	T_1	T_2	...	T_i	...	T_n
k_1	·	·	...	·	...	·
⋮	⋮	⋮	⋮	⋮	⋮	⋮
k_j	·	·	...	·	...	·
⋮	⋮	⋮	⋮	⋮	⋮	⋮
k_m	·	·	...	·	...	·

The 2-step approach

- a Neural Network approximates the map from model parameters to (reduced) volatility surfaces (**offline training**)

input	output
θ	$(\sigma_{BS}^{\mathcal{M}}(T_i, k_j, \theta))_{i=1,\dots,n \;\; j=1,\dots,m}$

- solve the optimization problem **online**

Calibration by feed-forward Neural Networks cnt'd

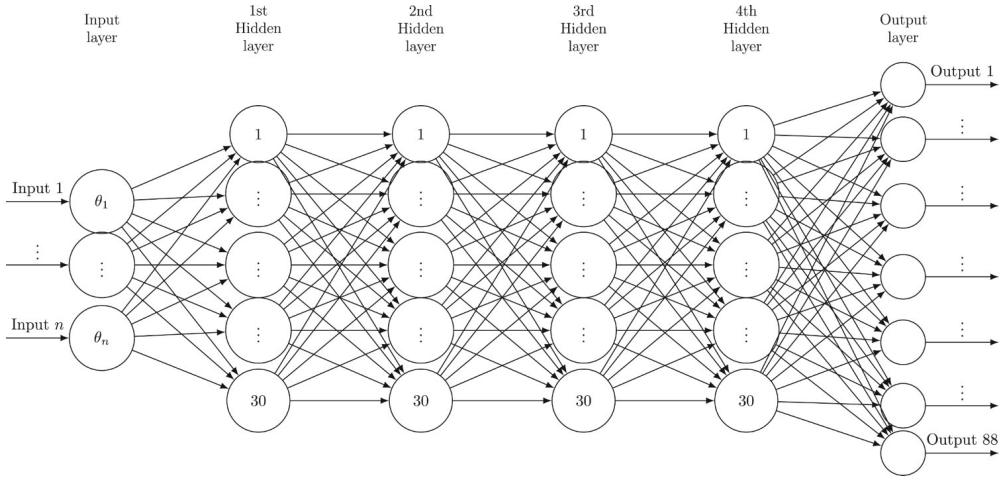


Figure: Neural network architecture in Horvath et al. (2021).

PROS **calibration is extremely fast!** One millisecond to compute one 11×8 grid of implied volatilities, nearly 100 steps to convergence

PROS Rømer's adaptive grid enhances the performances

CONS **need for an interpolation/extrapolation method to compute the implied volatility for arbitrary (k, T)**

CONS **extrapolation, expecially at very low time to maturity, may be subtle**

The pointwise approach

- A feed-forward neural network is trained to learn the pricing function from model parameters and option's parameters to (a single point in) implied volatility

input	output
(T, k, θ)	$\sigma_{BS}^{\mathcal{M}}(T, k, \theta)$

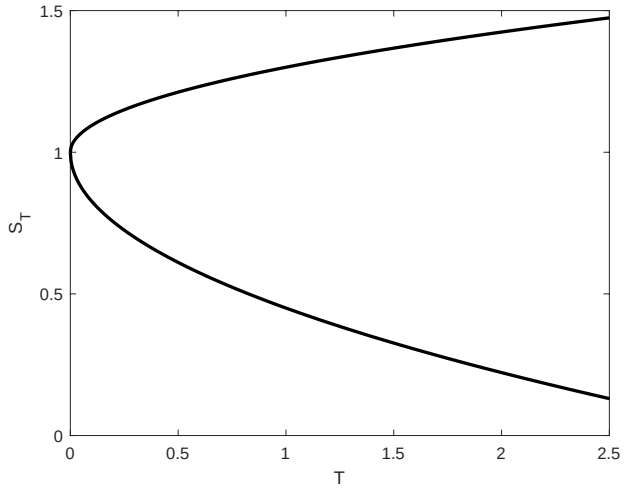
- The approach was pioneered by Bayer and Stemper (2018). **By design, it is interpolation-free.** Unfortunately, the architecture in the original specification was quite demanding: from few hundreds up to four thousands nodes per layer. The number of samples must be large for proper training.

Deep learning with random grids

Baschetti, Bormetti, and Rossi. *Quantitative Finance*, **24**(9):1263–1285, 2024

We identified three important ingredients to boost learning

1. An **adaptive time-to-maturity - moneyness grid**:



2. When required by the model, a low-dimensional parametric specification of the forward variance curve performs very well w.r.t. piecewise constant specifications, e.g., a Nelson-Siegel like form

$$\xi_0(t) = \beta_0 + \beta_1 \exp\left(-\frac{t}{\tau_1}\right) + \beta_2 \left(\frac{t}{\tau_2}\right) \exp\left(-\frac{t}{\tau_2}\right)$$

Deep learning with random grids cnt'd

3. **Pointwise training can be performed more effectively**: One produces **random grid surfaces** in the generation phase **but feed them to the network in a pointwise manner**

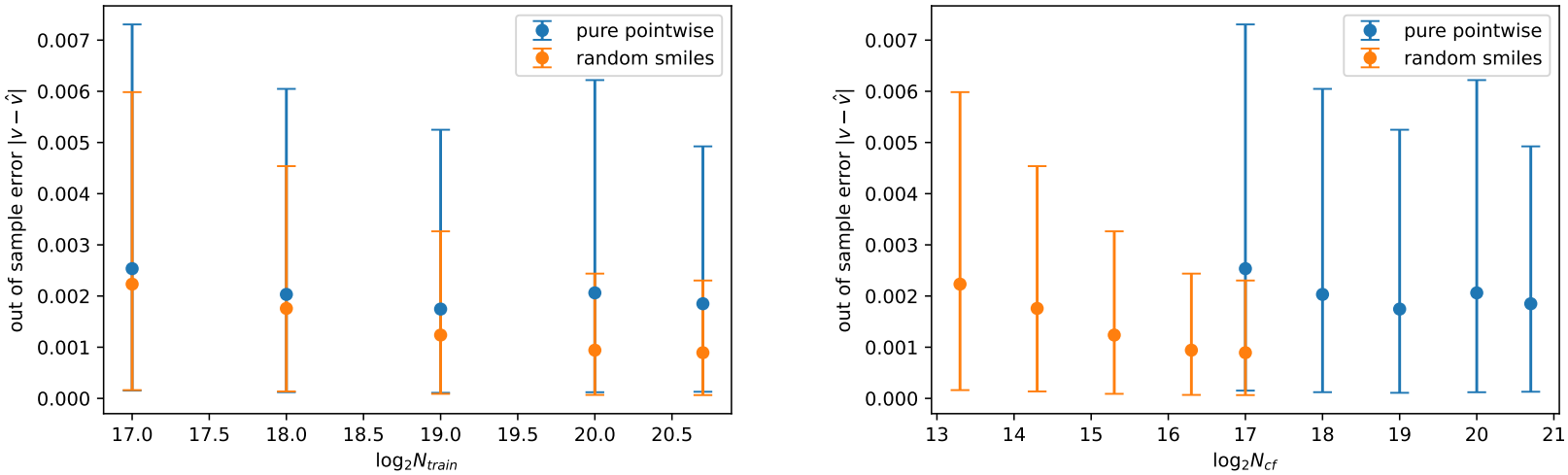


Figure: Out-of-sample errors as a function of the training sample dimension (left) and of the computational burden (right). Blue bars: pointwise training. Orange bars: random smiles training.

4. Network's architecture is as simple as Horvath et al (2018)

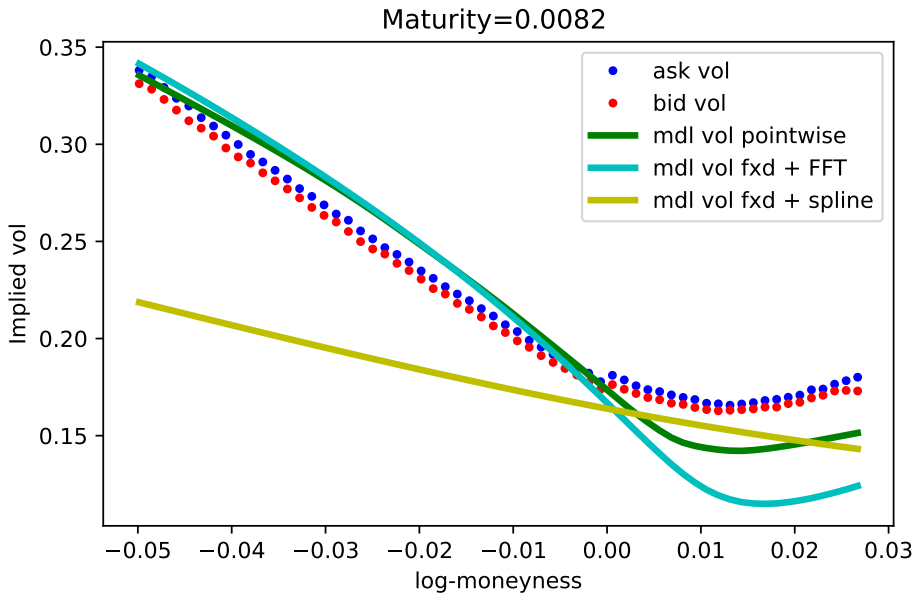


Figure: Rough Heston calibration to the market volatility surface as of March 16, 2021. Green is pointwise (random grid), cyan is fixed grid (interpolation via FFT of the true CF), and yellow is fixed grid plus splines.

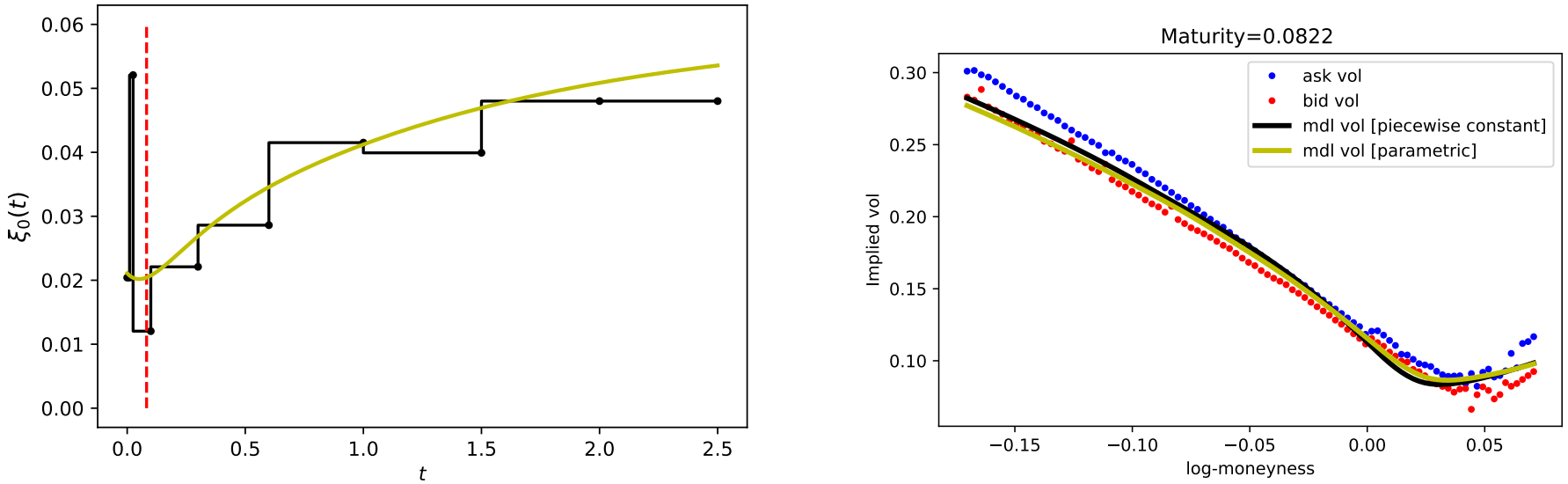


Figure: Left panel: Calibrated piecewise constant (black) and parametric (yellow) forward variance curve as of 2014/07/17. Right panel: fit to the $T = 0.0822$ maturity smile.

II Part

Joint SPX/VIX calibration of the 4-factor PDV Markov model of Gazzani & Guyon (2024)

Pricing and calibration in the 4-factor PDV model

Gazzani & Guyon. *To appear in Quantitative Finance*, 2024

Double exponential kernel specification (to mimic **volatility persistence** via *heterogeneous time scales*)

$$K_{\theta,\lambda_0,\lambda_1}(\tau) : \tau \mapsto (1-\theta)\lambda_0 \mathrm{e}^{-\lambda_0 \tau} + \theta\lambda_1 \mathrm{e}^{-\lambda_1 \tau} \quad \text{where } \theta \in [0,1],\ \lambda_0 > \lambda_1 > 0$$

Factors decompose as

$$\begin{aligned} R_{1,t} &= (1-\theta_1) R_{1,0,t} + \theta_1 R_{1,1,t} \\ R_{2,t} &= (1-\theta_2) R_{2,0,t} + \theta_2 R_{2,1,t} \end{aligned}$$

with dynamics

$$\begin{aligned} \mathrm{d}R_{1,j,t} &= \lambda_{1,j} \left(\sigma(R_{1,t}, R_{2,t}) \mathrm{d}W_t - R_{1,j,t} \mathrm{d}t \right) \\ \mathrm{d}R_{2,j,t} &= \lambda_{2,j} \left(\sigma(R_{1,t}, R_{2,t})^2 - R_{2,j,t} \right) \mathrm{d}t \qquad j \in \{0,1\}. \end{aligned}$$

Model parameters:

$$\boldsymbol{\Theta} = (\underbrace{\beta_0, \beta_1, \beta_2, \beta_{1,2}, \lambda_{1,0}, \lambda_{1,1}, \theta_1, \lambda_{2,0}, \lambda_{2,1}, \theta_2}_{\boldsymbol{\theta}}, \underbrace{R_{1,0,0}, R_{1,1,0}, R_{2,0,0}, R_{2,1,0}}_{\boldsymbol{R}_0}).$$

VIX pricing

VIX in a Markov setting

Within a Markovian pricing model (with continuous paths)

$$VIX_T^2(\boldsymbol{\theta}, \mathbf{R}_T) = \frac{1}{\Delta}\mathbb{E}\left[\int_T^{T+\Delta} \sigma_t^2 dt \middle| \mathbf{R}_T\right]$$

In Gazzani & Guyon (2024)

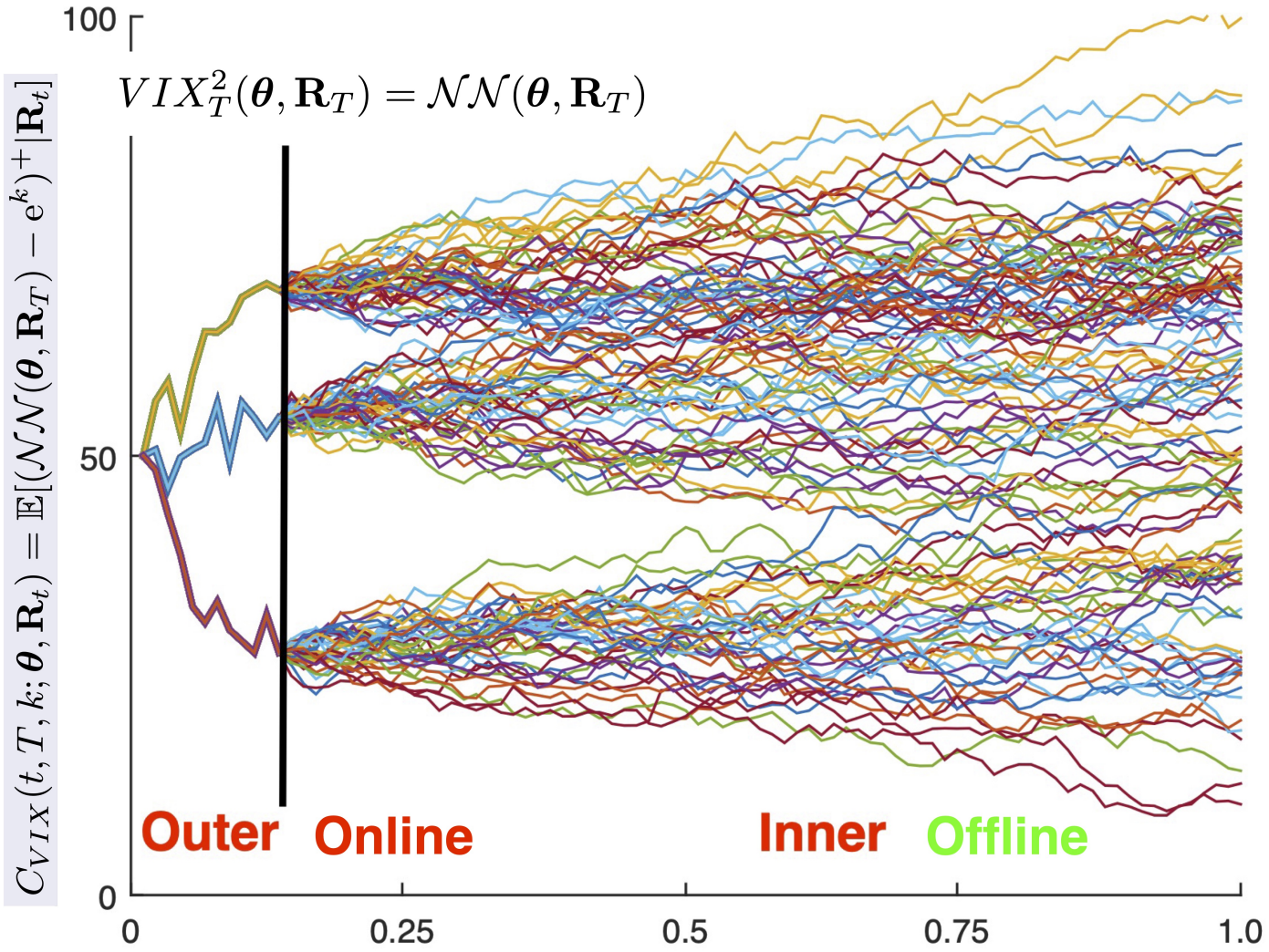
$$VIX_T^2(\boldsymbol{\theta}, \mathbf{R}_T) = \frac{1}{\Delta}\mathbb{E}\left[\int_T^{T+\Delta} (\beta_0 + \beta_1 R_{1,t} + \beta_2 \sqrt{R_{2,t}} + \beta_{1,2} R_{1,t}^2 \mathbf{1}_{R_{1,t}>0})^2 dt \middle| \mathbf{R}_T\right]$$

VIX Derivatives

VIX Futures and Options

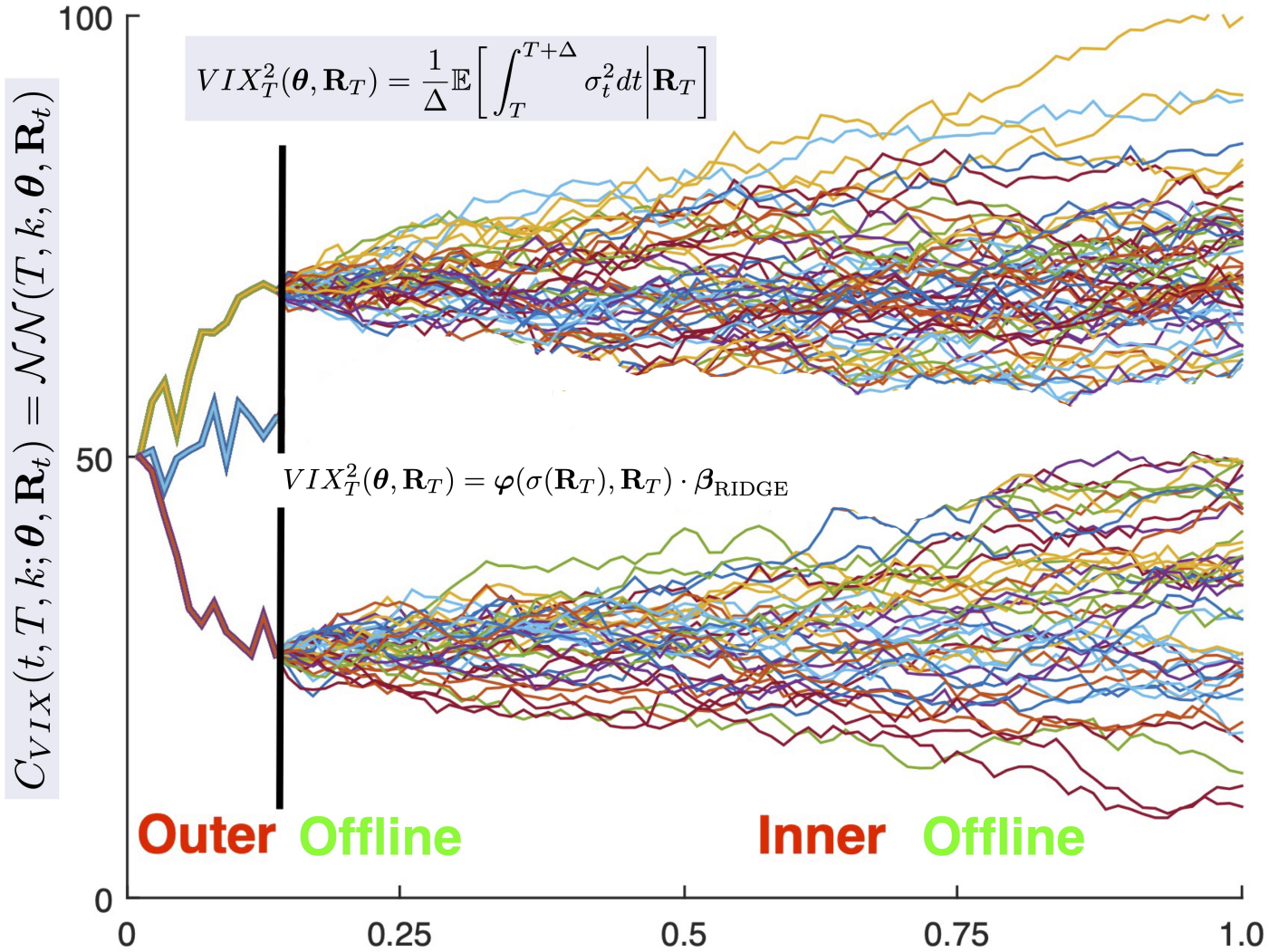
$$F_{VIX}(t, T; \boldsymbol{\theta}, \mathbf{R}_t) = \mathbb{E}[VIX_T | \mathbf{R}_t]\,,$$
$$C_{VIX}(t, T, k; \boldsymbol{\theta}, \mathbf{R}_t) = \mathrm{e}^{-r(T-t)}\mathbb{E}[(VIX_T - \mathrm{e}^k)^+ | \mathbf{R}_t]\,.$$

Hybrid Monte Carlo - Neural Networks

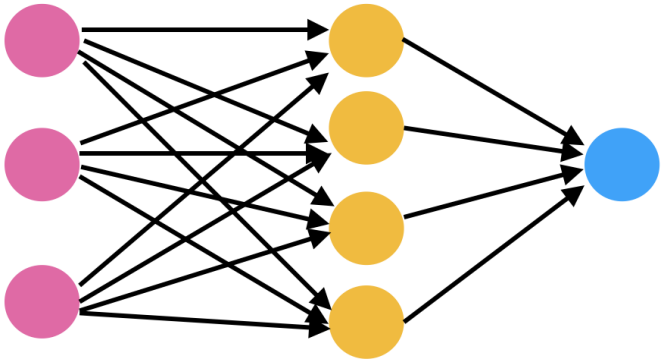


Our contribution in a nutshell

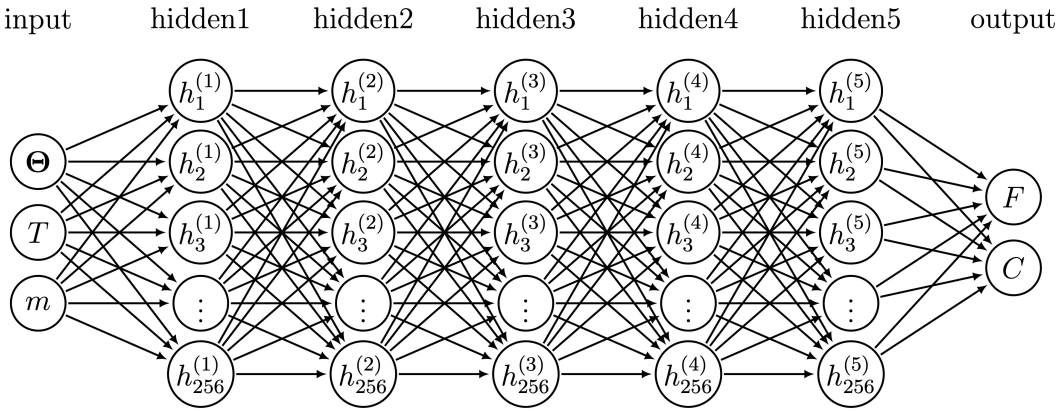
Baschetti, Bormetti, and Rossi. *In preparation*, 2025



Joint neural calibration



$$(T, m, \boldsymbol{\Theta}) \mapsto IV_{SPX}$$

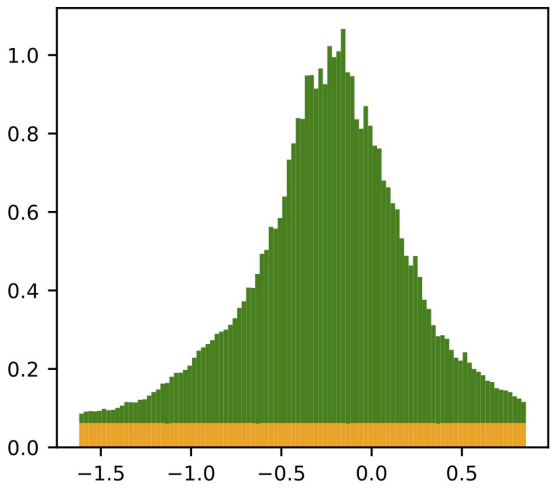


$$(T, m, \boldsymbol{\Theta}) \mapsto (F_{VIX}, C_{VIX})$$

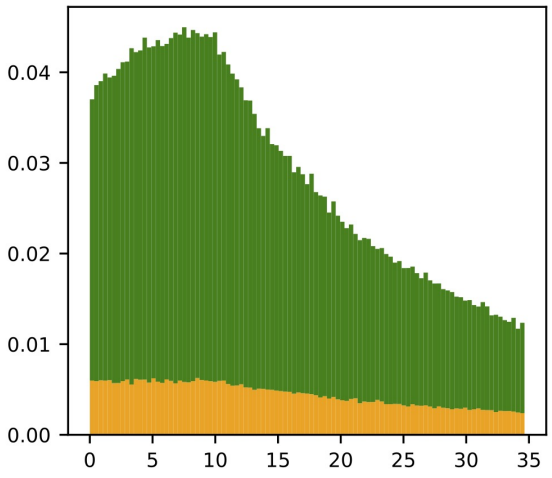
- Neural nets on SPX learn Implied Volatilities (IVs)
- Neural nets on VIX learn **futures and option prices, jointly** (no IVs on futures, joint learning enforces consistency)

Key aspects in the learning phase

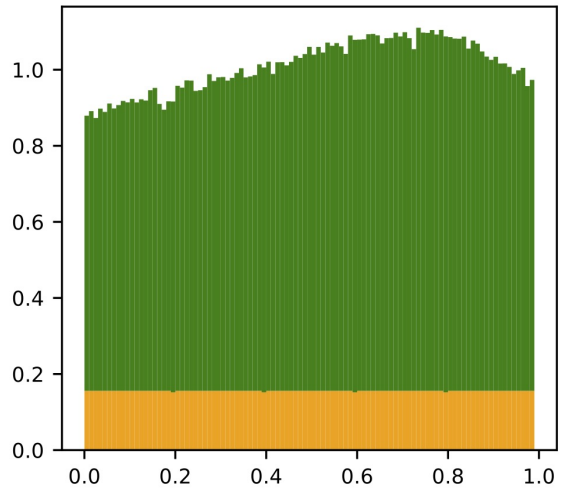
- **Full Nested Monte Carlo benchmark**: # outer paths $= 2^{18}$, # inner paths $= 2^{13}$;
  Nested Monte Carlo plus ridge Least Squares: # outer paths $= 2^{13}$, # inner paths $= 2^{10}$.
- **Not uniform** parameter sampling



R_{01}



λ_{11}



θ_1

- **Grid randomization**

Objective function

We opt for the following

$$
\begin{aligned}
L^J = {} & w_{vSPX} \frac{1}{N_T^{SPX}} \sum_{i=1}^{N_T^{SPX}} \left[\frac{1}{N_{K(T_i^{SPX})}} \sum_{l=1}^{N_{K(T_i^{SPX})}} \left(\frac{\sigma_{mdl}^{SPX}(T_i^{SPX}, K_l(T_i^{SPX}))}{\sigma_{mkt}^{SPX}(T_i^{SPX}, K_l(T_i^{SPX}))} - 1 \right)^2 \right] + \\
& + w_{fVIX} \frac{1}{N_T^{VIX}} \sum_{i=1}^{N_T^{VIX}} \left(\frac{F_{mdl}^{VIX}(T_i^{VIX})}{F_{mkt}^{VIX}(T_i^{VIX})} - 1 \right)^2 + \\
& + w_{vVIX} \frac{1}{N_T^{VIX}} \sum_{i=1}^{N_T^{VIX}} \left[\frac{1}{N_{K(T_i^{VIX})}} \sum_{l=1}^{N_{K(T_i^{VIX})}} \left(\frac{\sigma_{mdl}^{VIX}(T_i^{VIX}, F_{mdl}^{VIX}(T_i^{VIX}), K_l(T_i^{VIX}))}{\sigma_{mkt}^{VIX}(T_i^{VIX}, F_{mkt}^{VIX}(T_i^{VIX}), K_l(T_i^{VIX}))} - 1 \right)^2 \right],
\end{aligned}
$$

where

- $w_{vSPX},\ w_{fVIX},\ w_{vVIX} \in \mathbb{R}$ weight the three contributions above
- $N_T^{SPX/VIX}$ is the number of SPX/VIX maturities
- $N_{K(T_\cdot^{SPX/VIX})}$ is the number of strikes for a given SPX/VIX maturity.

Calibration on October 21, 2009

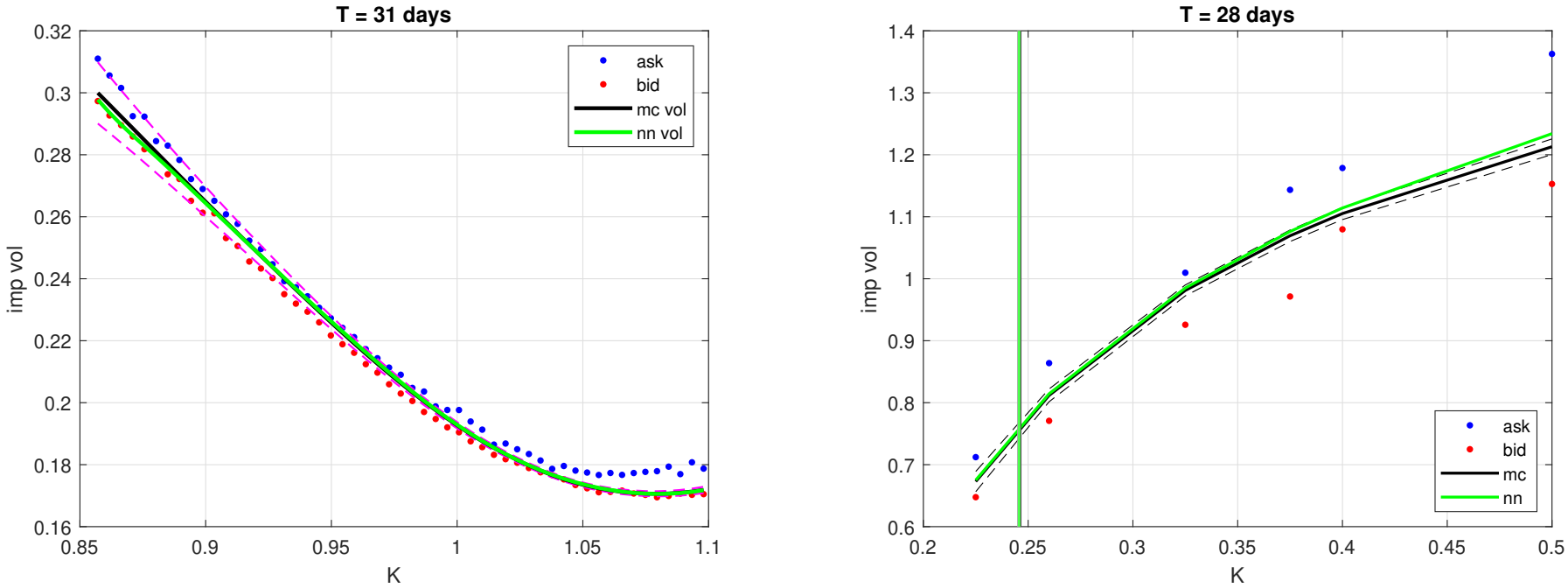


Figure: NN fit in green. Monte Carlo benchmarks (95% CB) in black (dashed).

β_0	β_1	β_2	$\beta_{1,2}$	$\lambda_{1,0}$	$\lambda_{1,1}$	θ_1	$\lambda_{2,0}$	$\lambda_{2,1}$	θ_2
0.0840	-0.2568	0.7415	0.2078	35.57	6.99	0.8142	10.15	0.21	0.9691

Table: Initial values of the factors: $R_{1,0,0} = 0.2261$, $R_{1,1,0} = 0.4361$, $R_{2,0,0} = 0.0281$, $R_{2,1,0} = 0.0460$. Futures prices: $F_{nn} = 0.2456$ *vs* $F_{mc} = 0.2461$.

Conclusions and perspectives

- Leveraging the pointwise approach to calibration, one can achieve
  - highly accurate pricing
  - fast computation of the implied volatility and sensitivities
  - **no need for interpolation and extrapolation**
  - **key ingredients**: **adaptive grid and random training with smiles**
- We provide a deep neural network pricer for SPX options and VIX derivatives that can be used for **real-time calibration** to market data
  - **key point**: **within the inner loop, generalization through ridge Least Squares**
- Extension to different specifications of the PDV model

Appendix

Learning phase

Sampling is not uniform in parameter space. The initial values of the factors are linked to the λ's via the past history of the index. Operatively,

$$R_{n,j,0} = \lambda_{n,j} \sum_{i=0}^{T-2} \mathrm{e}^{-\lambda_{n,j} \frac{i}{252}} \left(\frac{S_{-i}}{S_{-i-1}} - 1 \right)^n \qquad n = \{1, 2\}, \quad j = \{0, 1\}$$

where $T = 1008$ and S_0 is the index level from a date randomly sampled within the interval January 01, 2009 and December 31, 2023.

Learning phase cnt'd

Parameter	**Bounds**
θ_1	$[0, 1]$
θ_2	$[0, 1]$
$\lambda_{1,0}$	$[10, 65]$
$\lambda_{1,1}$	$[0, 35]$
$\lambda_{2,0}$	$[0, 50]$
$\lambda_{2,1}$	$[0, 15]$
$R_{1,0,0}$	$[-1.62, 0.88]$
$R_{1,1,0}$	$[-1.05, 0.71]$
$R_{2,0,0}$	$[0, 0.11]$
$R_{2,1,0}$	$[0, 0.11]$
β_0	$[0, 0.85]$
β_1	$[-0.30, -0.10]$
β_2	$[0.35, 0.95]$
$\beta_{1,2}$	$[0.05, 0.40]$

Table: Model parameters' bounds during the learning phase.