Solving The Dynamic Volatility Fitting Problem:

A Deep Reinforcement Learning Approach in Continuous State Action Spaces

Flow Equity Derivatives.

Emmanuel Gnabeyeu[1,2] *and Omar Karkar*[2]

October 14, 2025

[1]Ecole Polytechnique, Paris, France.
[2]EMEA Head of EQA, Citigroup Global Markets Limited, London, UK.

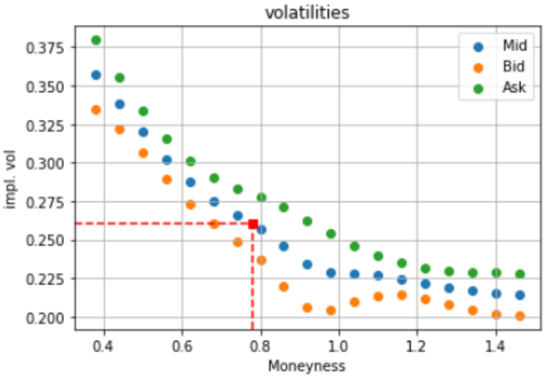
Table of Contents

Volatility Fitting Problem and why ?

Mark-to-market quoted price P: $\left\{P^{Ask}_{Call/Put}, P^{Bid}_{Call/Put}\right\} \xrightarrow{\phi^{\text{Model}}} \left\{\sigma^{Ask}_{Call/Put}, \sigma^{Bid}_{Call/Put}\right\} \xrightarrow{\phi} \left\{\sigma^{Ask}, \sigma^{Bid}\right\}$ for different moneyness $(\kappa_1, ..., \kappa_n)$ where ϕ is a non unique mapping and $\phi^{\text{Model}}=$ Black-Scholes for Ex.

Volatility Fitting Problem and why ?

Mark-to-market quoted price P: $\left\{P^{Ask}_{Call/Put}, P^{Bid}_{Call/Put}\right\} \xrightarrow{\phi^{\text{Model}}} \left\{\sigma^{Ask}_{Call/Put}, \sigma^{Bid}_{Call/Put}\right\} \xrightarrow{\phi} \left\{\sigma^{Ask}, \sigma^{Bid}\right\}$ for different moneyness $(\kappa_1, \ldots, \kappa_n)$ where ϕ is a non unique mapping and $\phi^{\text{Model}} =$ Black-Scholes for Ex.



The volatility fitting problems aims to match the market implied volatility. It is used:

- To price structured products with volatility exposure.
- For risk management
- etc.

Encoding the market data:

Many methods in the industry to encode the implied volatility information [3]

- Non-Parametric grid of impl. Vol then interpolation via simple spline.
- Modelling the implied density of the asset directly (Carr and Wu approach)
- Use a diffusion style model ((L)SV models)

[3]flexible and compatible with arbitrage constraints (call spread, butterfly and calendar spread)

[4]Ex: Stochastic volatility inspired (SVI) parameterization by Gatheral and Antoine Jacquier: $\vec{\theta} = (a, b, \rho, m, \sigma)$

$$\Psi_{\vec{\theta}}^{SVI}(k)^2 = a + b(\rho(k - m) + \sqrt{(k - m)^2 + \sigma^2})$$

Encoding the market data:

Many methods in the industry to encode the implied volatility information [3]

- Non-Parametric grid of impl. Vol then interpolation via simple spline.
- Modelling the implied density of the asset directly (Carr and Wu approach)
- Use a diffusion style model ((L)SV models)
- Surface-Level Parametrizations [4]

$\rightarrow$ In this setting, we model the vol with a parametric functional $\Psi_{\vec{\theta}}^{vol}(\kappa)$ for a given maturity.

---

[3] flexible and compatible with arbitrage constraints (call spread, butterfly and calendar spread)

[4] Ex: Stochastic volatility inspired (SVI) parameterization by Gatheral and Antoine Jacquier: $\vec{\theta} = (a, b, \rho, m, \sigma)$

$$\Psi_{\vec{\theta}}^{SVI}(k)^2 = a + b(\rho(k-m) + \sqrt{(k-m)^2 + \sigma^2})$$

Traditional approach to volatility fitting with Parametrisation:

$\Rightarrow$ Choice of the target objective function:[5] : Among Others, the squared-error loss

$$\xi(\vec{\theta}_{t_i}) := \sqrt{\frac{\sum_{j=1}^{n} w_j \ell(t_i, \kappa_j)^2}{\sum_{j=1}^{n} w_j}} \text{ with } \ell(t_i, \kappa_j) := \frac{\sigma^{Mid}(t_i, \kappa_j) - \Psi^{vol}_{\vec{\theta}_{t_i}}(\kappa_j)}{\sigma^{\text{spread}}(t_i, \kappa_j)}, \quad w_j = \text{Black-Scholes Vega} \tag{1}$$

$\Rightarrow$ Given vol. surf. param. $\Psi_{\vec{\theta}_{t_i}}$ and $\xi(\vec{\theta}_{t_i})$, adjust $\vec{\theta}_{t_i}$ to match the mid market observable impl. vol. σ^{Mid}

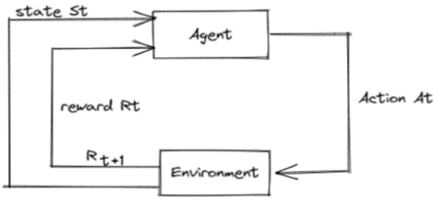
Drawbacks:

- Large set of conditions (liquidity, macro-events, term-structure, stability)$\rightarrow$ optimizer too heavy to run
- Imply an optimisation at each move of the market (speed and reliability).
- Cannot re-use past accumulated experiences (every output is flashed out).
- Do not handle very well a case with market data limitations and irregularities.

$\Rightarrow$ We rely on a fully AI-based method to learn market regimes and discover optimal behaviours.

[5] Practitioners aim at positioning the model implied volatility at the mid.

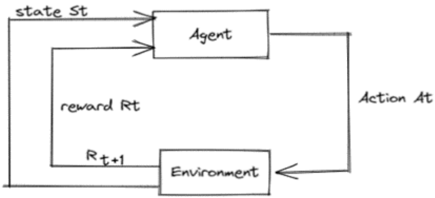
Towards a Novel Framework: Background on Reinforcement Learning.



An agent reinforced with awards based on its interaction with the environment would learn to operate optimally in that environment to maximize the rewards.

- The agent observes a current state $s_t \in \mathcal{S} \subset \mathbb{R}^d$, takes an action $a_t \in \mathcal{A} \subset \mathbb{R}^n$ drawn from the policy $\Pi \supset \pi : \mathcal{S} \to \mathcal{A}$ (resp. $\mathcal{P}(\mathcal{A})$) and receives a reward $r_t = r(a_t, s_t)$.

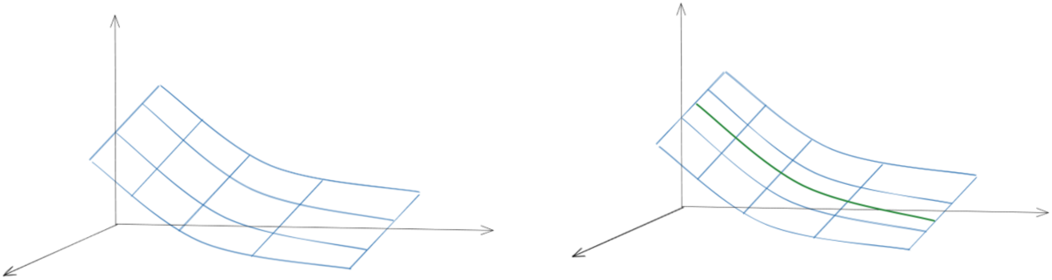
Towards a Novel Framework: Background on Reinforcement Learning.



An agent reinforced with awards based on its interaction with the environment would learn to operate optimally in that environment to maximize the rewards.

- The agent observes a current state $s_t \in \mathcal{S} \subset \mathbb{R}^d$, takes an action $a_t \in \mathcal{A} \subset \mathbb{R}^n$ drawn from the policy $\Pi \supset \pi : \mathcal{S} \to \mathcal{A}$ (resp. $\mathcal{P}(\mathcal{A})$) and receives a reward $r_t = r(a_t, s_t)$.
- Set $\gamma \in [0,1]$, define the Return from a state from following the policy π, $R_t = \sum_{i=t}^{T} \gamma^{(i-t)} r(s_i, a_i)$
- state-action value function $Q^\pi(s,a) = \mathbb{E}^\pi[R_t | s_t = s, a_t = a]$.
- Mathematical formulation of the problem: $\boxed{\pi^* = \arg\max_{\pi \in \Pi} Q^\pi}$

Problem Statement in Reinforcement Learning:

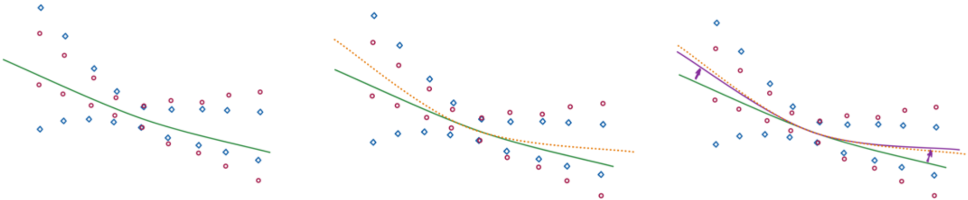


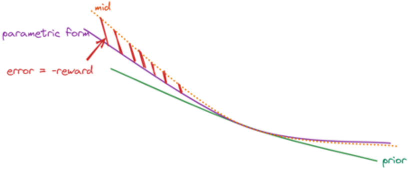
(a) (b)

- Fixed expiry T, and observe the market at time t_i, i.e. a collection $\left\{\sigma^{Ask}_{Call/Put}(t_i,\kappa_j),\sigma^{Bid}_{Call/Put}(t_i,\kappa_j)\right\}_{j\in[1,\ldots,n]}$ of markets quotes for different moneyness $(\kappa_1,\ldots,\kappa_n)$.
- we consider a parametrisation $\Psi^{vol}_{\vec{\theta}}$ of the volatility slice with a vector of K parameters.

RL Framework for Fitting:

- Starting from a prior parametrized volatility surface $\Psi^{vol}_{\vec{\theta}_{t_i}}$ (i.e. the latest fit is the prior input).





Advanced Fitting: a reinforcement learning approach

Goodness of Fit in RL-Based Volatility Fitting

- Mean squared error (MSE) [6] defined by :

$$\xi(\vec{\theta}_{t_i}) := \sum_{j=1}^{n} \left(\sigma^{Mid}(t_i, \kappa_j) - \Psi^{vol}_{\vec{\theta}_{t_i}}(k_i) \right)^2 \tag{3}$$

- The problem is to find the shifted vector $\Delta\vec{\theta}_{t_i} = \left(\Delta\theta^1_{t_i}, \ldots, \Delta\theta^K_{t_i}\right)$ to bump the old parameters $\vec{\theta}_{t_i}$ in order to maximize our selected reward function. $\boxed{\Delta\vec{\theta}^*_{t_i} = \arg\max_{\Delta\vec{\theta}_{t_i} \in \mathbb{R}^K} Q^\pi(s_{t_i}, \Delta\vec{\theta}_{t_i}).}$

with $Q^\pi(s_{t_i}, \Delta\vec{\theta}_{t_i}) = \sum_{t=t_i}^{T} \mathbb{E}_{(s_t, a_t)\sim\rho_\pi}\left[\gamma^{(t-t_i)} r(s_t, \Delta\vec{\theta}_t)\right]$ and $\underline{r(s_t, a_t = \Delta\vec{\theta}_t) = -\xi(\theta_t)}$

[6] Alternative. Setting $\sigma^{\text{spread}}(t_i, \kappa_j) := \sigma^{Ask}_{t_i,\kappa_j} - \sigma^{Bid}_{t_i,\kappa_j}$ the volatility spread, we define:

$$\xi(\vec{\theta}_{t_i}) := \sum_{j=1}^{n} \text{vega}_{BS}(\kappa_j) \left(\frac{\sigma^{Mid}(t_i, \kappa_j) - \Psi^{vol}_{\vec{\theta}_{t_i}}(\kappa_j)}{\sigma^{\text{spread}}(t_i, \kappa_j)} \right)^2 \tag{2}$$

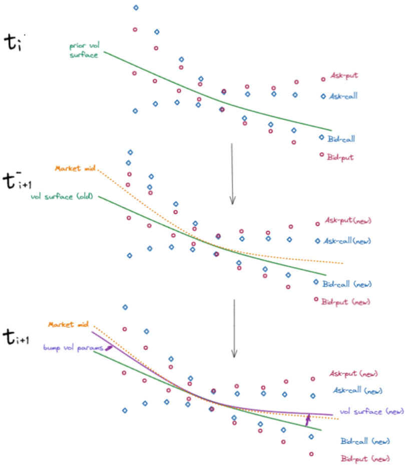
Market environment and Non Linear functional approximation by FCNNs:

1. States are the collection of observable market quotes:
   $s_{t_{i+1}} = \left(\sigma^{Bid}(t_{i+1}, \kappa_j), \sigma^{Ask}(t_{i+1}, \kappa_j), \theta^1_{t_i}, \dots, \theta^K_{t_i}\ \right)_j$
2. Actions are the bumps to apply in order to shift the prior volatility parametrization:
   $a_{t_{i+1}} = \Delta\vec{\theta}_{t_{i+1}} = \vec{\theta}_{t_{i+1}} - \vec{\theta}_{t_i}$
3. Rewards are the opposite of well designed errors.
   $r(s_{t_{i+1}}, a_{t_{i+1}} = \Delta\vec{\theta}_{t_{i+1}}) = -\xi(\theta_{t_{i+1}})$

$$Q^\pi(s_{t_{i+1}}, a_{t_{i+1}}) = -\sum_{k=t_{i+1}}^{T} \mathbb{E}_{(s_k, a_k)\sim\rho_\pi}[\gamma^{(k-t_{i+1})}\xi(\vec{\theta}_k)].$$

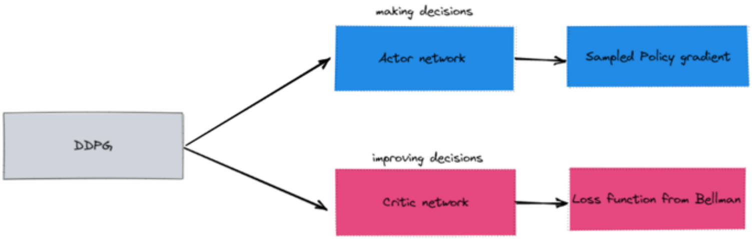
$$\boxed{a^*_{t_{i+1}}(s_{t_{i+1}}) = \arg\max_{a_{t_{i+1}}\in\mathcal{A}} Q^\pi(s_{t_{i+1}}, a_{t_{i+1}})}$$

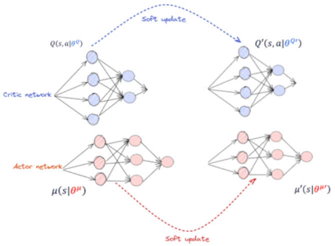
Remark: $\Delta\vec{\theta}$ (the action vectors) can take any values in $\mathbb{R}^K$ space: $\Rightarrow$ continuous actions spaces.



Actor Critic: Deep Deterministic Policy Gradient.(Lillicrap et al., 2015)

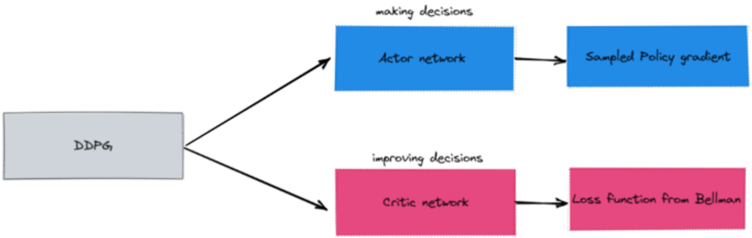
- off-policy actor-critic method that learns a deterministic policy in continuous domain.

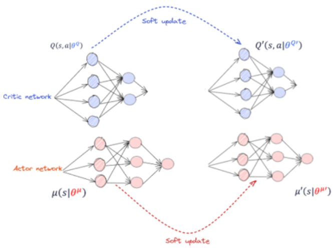




Actor Critic: Deep Deterministic Policy Gradient.(Lillicrap et al., 2015)

- off-policy actor-critic method that learns a deterministic policy in continuous domain.



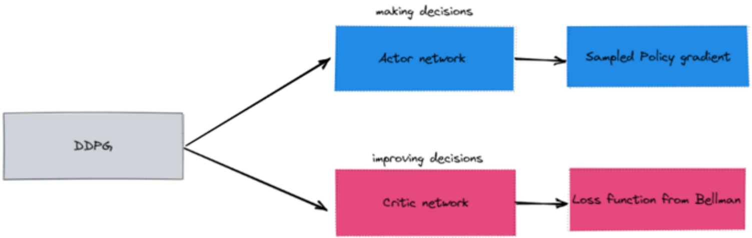


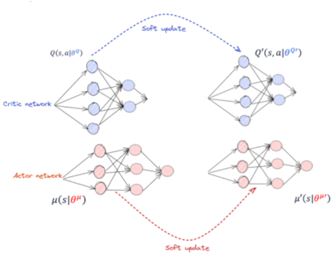
- Action network: DPG= Deep Policy gradient

$$a_t \sim \pi^D(s_t, \theta^\pi) + \epsilon_t \quad \text{with} \quad \epsilon_t \sim \mathcal{N}(0, \sigma_n^2 I_K) \quad \text{and} \quad \sigma_n = \max(\sigma_0(1 - \frac{n}{N})^4, \sigma_{\min})$$

Actor Critic: Deep Deterministic Policy Gradient.(Lillicrap et al., 2015)

- off-policy actor-critic method that learns a deterministic policy in continuous domain.





- Action network: DPG= Deep Policy gradient

$$a_t \sim \pi^D(s_t, \theta^\pi) + \epsilon_t \quad \text{with} \quad \epsilon_t \sim \mathcal{N}(0, \sigma_n^2 I_K) \quad \text{and} \quad \sigma_n = \max(\sigma_0(1 - \frac{n}{N})^4, \sigma_{\min})$$

- Critic network: Q-Learning and Bellman equation.

$$\begin{cases} R_t = \sum_{i=t}^{T} \gamma^{(i-t)} r(s_i, a_i) \\ Q^\pi(s_t, a_t) = \mathbb{E}[R_t | s_t, a_t] \end{cases} \Rightarrow L(\theta^Q) = \mathbb{E}[(Q^\pi(s_t, a_t; \theta^Q) - [r(s_t, a_t) + \gamma Q^\pi(s_{t+1}, a_{t+1}; \theta^Q)])^2]$$

Numerical results: Convergence of DDPG Algorithm.

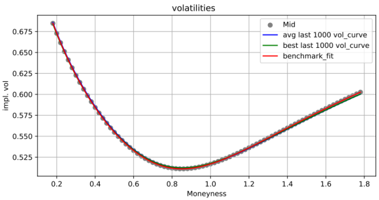
- To assess the performance of the RL algorithms for the volatility fitting, we consider toy problems of increasing complexity and perform a Monte Carlo on 5 differents random seeds:

1. Static scenario:
    - we explore the scenarios where the market conditions remain constant.

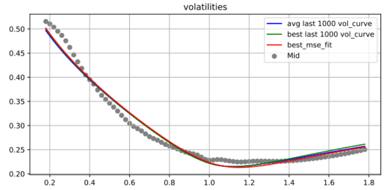
Play Video

Static Market, DDPG algorithm, MSE Reward:

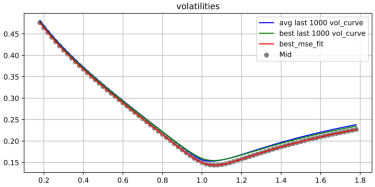
Below, best response of the agent amongst the last 1000 episodes (*blue*) and the mean of the last 1000 volatilities slices (*red*) (a) High Smile (b) Skew (c) High Skew (d) Inverse Smile.

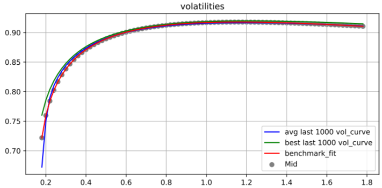


(a)



(b)





DDPG: Convergence and Consistency.

- Performance of the RL algorithms: Monte Carlo on 5 differents random seeds:
  1. Static scenario:
     - RL agents perform as-good as traditional approaches in fitting volatility curve.

Table: DDPG MSE Rewards

type	Skew	High Smile	Inv. Smile
Bench	-0.011913	-0.0000220	-0.0000436
seeds' avg	-0.012145	-0.000163	-0.000315

- Final rewards achieved is close to the one coming from the optimizer.
- Performance is stable across the different market configuration considerered

Quasi Dynamic Scenario: The market evolve freely.

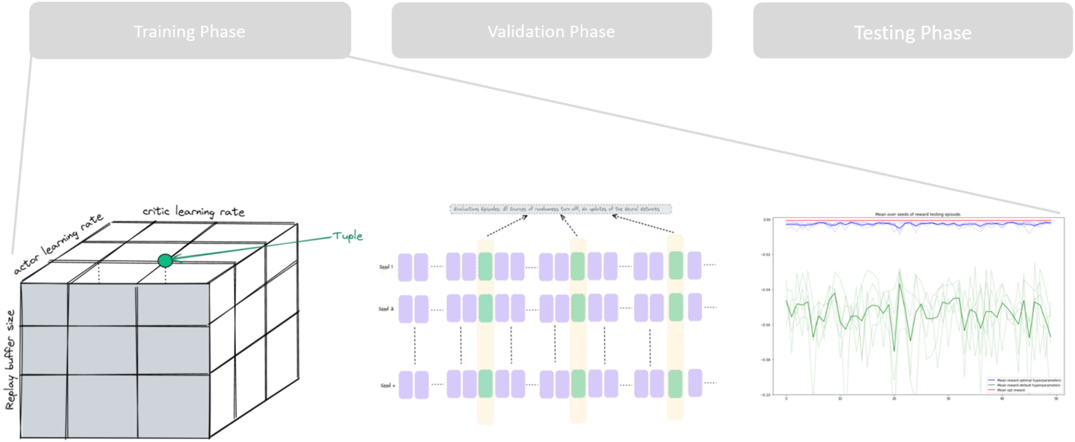


Figure: Training and Consistency over random seeds.

Quasi Dynamic Scenario, MSE Reward

2.Validation Phase: We compare the performance of different successful agents under different seeds.

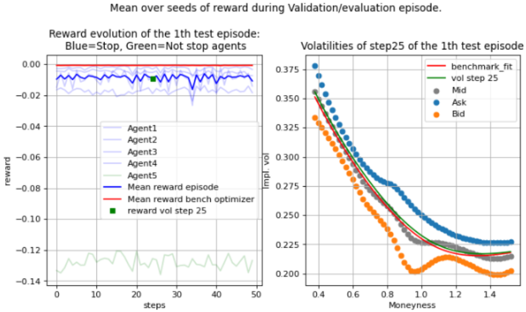


Figure: Evolution of mean episodic rewards(several random seeds) and Implied volatility(of one selected step) for a wide spread stock with **DDPG**.

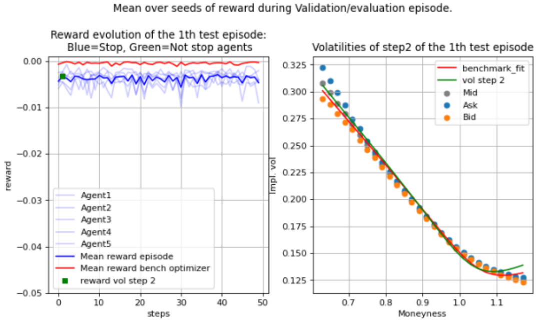


Figure: Evolution of mean episodic rewards(several random seeds) and Implied volatility(of one selected step) for a tight spread stock with **DDPG**.

Quasi Dynamic Scenario, MSE Reward, Testing Phase

3.Testing Phase: we assess the performance of the best agent in a test episode.

Play Video

Vol fitting animation

Conclusion and Prospects

1. Sum up,
   - A wide range of problems can be tackled with Deep RL Actor-Critic methods while neural networks help to approximate the solution function.[7]
   - We showed that DRL algorithms are natively tailored for the volatility fitting problem:
     - native exploration
     - effective catalog of past experiences(re-use past accumulated experiences)
     - handle complex objective function (desk PnL).

[7] optimal execution, portfolio optimization, option pricing and hedging, market making, smart order routing

Conclusion and Prospects

1. Sum up,
   - A wide range of problems can be tackled with Deep RL Actor-Critic methods while neural networks help to approximate the solution function.[7]
   - We showed that DRL algorithms are natively tailored for the volatility fitting problem:
     - native exploration
     - effective catalog of past experiences(re-use past accumulated experiences)
     - handle complex objective function (desk PnL).

2. Future work could be:
   - Reflects stylistic effects in volatility surface term-structure.
   - Careful rewards shaping and environment design.

[7] optimal execution, portfolio optimization, option pricing and hedging, market making, smart order routing

Thanks for your attention!

References.

1. B. Hambly, R. Xu, and H. Yang, Recent Advances in Reinforcement Learning in Finance.
2. J. Gatheral, A. Jacquier. Arbitrage-Free SVI Volatility Surfaces
3. V. Zetocha, Sculpting implied volatility surfaces of illiquid assets
4. Sutton, R. S. and Barto, A. G, Reinforcement learning: An introduction.
5. T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra,Continuous control with deep reinforcement learning
6. R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, Policy gradient methods for reinforcement learning with function approximation