Reinforcement Learning under the Avellaneda Stoikov Theoretical Market Making Framework

Jadon Ng Tsz Hei
The University of Hong Kong, Quantitative Finance and Computer Science

ABSTRACT: This study explores the application of reinforcement learning (RL) to the optimal market-making problem within a theoretical framework. The goal of the study is to evaluate the effectiveness of various RL algorithms in enhancing market-making strategies and compare their performance against established benchmarks in existing literature. By simulating a controlled trading environment, the adaptability and efficiency of RL-based approaches in managing inventory risk and maximizing profit is analyzed. At the same time, it also aims to provide insights into the potential advantages and limitations of incorporating reinforcement learning techniques in market-making. This study serves as an extension to (Selser et al., 2021) work by simulating on a wider variety of reinforcement learning algorithms as well as analyze the impact on profit capturing and inventory management ability of different commonly used reward functions in existing literature.

1. Introduction

In recent years, the application of reinforcement learning in the financial industry has gain significant attention due to its potential to enhance trading strategies, risk management, and decision-making processes. Reinforcement learning techniques, inspired by the way humans learn through trial and error, have shown promise in adapting to dynamic and complex financial markets. These approaches enable autonomous agents to learn optimal strategies by interacting with the environment, receiving feedback in the form of rewards or penalties based on their actions. In the context of the financial industry, reinforcement learning algorithms have been utilized to optimize trading execution, portfolio management, and market making strategies.

In this study, classical benchmarks such as symmetrical quoting and the Avellaneda Stoikov model are leveraged as benchmarks to compare their performance with different reinforcement learning methods. This study will also analyze how different reward functions and changes to hyperparameter will affect the learning efficiency as well as how will they change the agent's behaviours in terms of their inventory control and spread capturing abilities.

2. Preliminaries

2.1. Market Making

The role of a market maker is to provide market liquidity through constantly quoting on both side of the market and is compensated from the bid-ask spread on the limit order book given that the market maker place bid (buy) orders at a lower price from his/her ask(sell) orders. This process of actions is known as market making.

The profit of a market maker is directly proportional to the number of bid and ask orders matched by the market, that is, how frequently a market maker is able to quote on both side of the market and how frequently is the orders being executed in the limit order book.

A market maker is mainly exposed to two risks, the *inventory risk* and *adverse selection risk*. While a market maker continuously profits from the seeminglessly riskless bid-ask spread, the market maker will be exposed to the risk of accumulating net inventory if only one side of market maker's order is being matched. The accumulation of large inventory exposes the market maker to huge loss if the market goes to the opposite side of the inventory. Adverse selection risk is the risk where your orders are being matched by an informed buyer/seller, thereby accumulating inventory opposing the market direction as a result.

In general, a market maker's goal is to place the bid and ask orders on the market strategically to profit from bid-ask spread while minimizing the inventory risk and adverse selection risk by market movements.

2.2. The Avellaneda Stoikov Model

The Avellaneda Stoikov Model introduces a classical modelling methodology of market making environment as a stochastic optimal control problem to solve for the optimal prices to place the bid and ask orders for high frequency market making (Avellaneda & Stoikov, 2008).

Compare to a traditional market making benchmark to place bid and ask orders symmetically around market midprice, AS model consider multiple key variables for placing bid and ask orders for each tick including current market maker's inventory, market volatility, time until terminal (market closes) alongside order book statistics such as market volatility and order book intensity under certain modelling assumptions (Section 3).

The idea behind AS model is to solve for two variables, reserved price and optimal spread. Instead of placing symmetrical bid ask orders around the market's mid price, AS model places symmetrical bid and ask orders around the reserved price with width equal to the optimal spread in a way that considers the aforementioned variables to minimize inventory risk and adverse selection risk.

Full code implementation can be found at https://github.com/crxxom/RL-AS

The reserved price is modelled as,

$$r(s,q,t) = s - q\gamma\sigma^2(T-t)$$

where,

s: $market\ mid\ price$
q: $market\ maker's\ current\ inventory$
σ: $market\ volatility$
γ: $risk\ adversion\ parameter$
$T-t$: $time\ until\ market\ close/Terminal\ time$

And the optimal spread is modelled as,

$$\delta^a + \delta^b = \gamma\sigma^2(T-t) + \frac{2}{\gamma} ln\left(1 + \frac{\gamma}{\kappa}\right)$$

where κ represents the order book intensity.

A market maker would then place the optimal bid/ask limit orders on $p^b = r_t - \frac{\delta}{2}, p^a = r_t + \frac{\delta}{2}$ to perform high frequency market making.

Under the equations, one may deduce the following relations between the optimal bid ask quotes and current market maker's state.

1. The higher the *inventory* is, the lower is the reserved price, that is, the lower the quoted ask price to sell out the inventory, vice versa

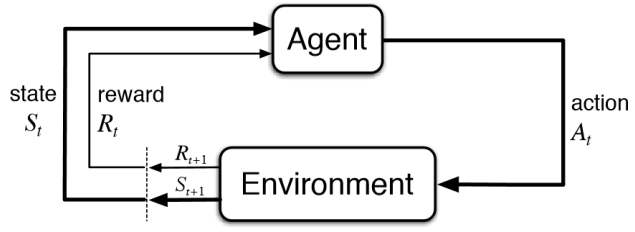
2. The closer to *terminal time*, the smaller is the optimal spread and deviation to market's mid price, vice versa

3. The higher the *market's volatility* is, the larger is the optimal spread and deviation to market's mid price, vice versa

4. The larger the *order book intensity* is, the smaller is the market spread and optimal spread.

2.3. Reinforcement Learning

Reinforcement learning is one of the three areas under machine learning alongside supervised learning and unsupervised learning. It is a computation way to learn through the interaction with the environment and quantize learning to an optimization problem of maximizing expected numerical reward signal.



Source: Medium.com

The above diagram illustrates the typical interation framework of a reinforcement learning environment.

At time t, the *agent* recieves a *state* S_t and output a corresponding action A_t to be taken, this process of mapping a state to an action is called the *policy* of the algorithm. The action A_t taken will then interact with the environment and return a new state S_{t+1} and the *reward* R_{t+1} as a feedback signal for the taken action A_t. The process continues until it reaches the *terminal* state under a finite horizon reinforcement learning problem, which will be the focus of this study.

A reinforcement learning problem is often formally described as solving the *Markov Decision Process (MDP)* problem, or in some cases, a *Partially Observable Markov Decision Process (POMDP)*. Given any state and action, the probability of each possible pair of next state and reward is denoted by (Richard & Andrew, 2014),

$$p(s', r|s, a) = Pr\{S_{t+1} = s', R_{t+1} = r'|S_t = s, A_t = a\}$$

This translates the problem to optimizing the expected *discounted return* through policy π, where one can use a *state-value* function $V^*(s)$ to represent the expected future returns given a state.

$$V^*(s) = \sup_{\pi} V^{\pi}(s) = \sup_{\pi} E^{\pi}\left[\sum_{t=0}^{t=\infty} \gamma^t r(s_t, a_t|s_0 = s)\right]$$

where γ represents the discount factor and s_{t+1} follows the *state-transition probabilities*,

$$p(s'|s, a) = Pr\{S_{t+1} = s'|S_t = s, A_t = a\} = \sum_{r\in R} p(s', r|s, a)$$

and a_t follows $\pi_t(s_t)$, mapping current state to an action through an either determinisitics or randonmized policy π_t.

An example of policy is adopting a greedy algorithm and choose the action that maximize the *state-action value* given state s_t.

$$\pi(s) = \underset{a}{argmax}\ q_{\pi}(s, a)$$

One problem with this approach is that the greedy policy does not allow the agent to explore and may result in suboptimal policy. This problem is known as the *exploration and exploitation dilemma*. One solution to this problem is using a ϵ-greedy policy, whereas there is a probability ϵ that the policy will choose a randomize action that is not action that yield maximal value.

Although the above equation involves the state-transition probabilities, not all reinforcement learning algorithms requires the solving of state-transition probabilities. Algorithms that attempt to learn the underlying dynamics of the environment and solve for the state-transition probabilites are called *model-based* algorithms, while algorithms that learns the action-to-return associations through interaction trajectories without learning the underlying dynamic of the environment are called *model-free* algorithm.

Indeed, the goal of all reinforcement learning algorithms are essentially solving for the optimal policy π_*, which usually involves in solving the *state-value function* $v_*(s)$ or state-*action value function* $q_*(s, a)$ with traditional tabular learning using hashtables to store the value for all state-action

pairs or *deep reinforcement learning* methods that adapt different variations of artificial neural networks to achieve functional approximation. The goal of reinforcement learning is and maximize the expected reward through continuously interacting with the environment and eventually, allowing the received reward converge thereby returning an optimal policy for making a process of decisions to be taken.

3. Environment Assumptions

The study follows the underlying framework and assumptions made by the classical Avellaneda Stoikov market making model (Avellaneda & Stoikov, 2008). In particular, the following key assumptions are made.

1. Stock price evolves according to the driftless Brownian Motion

$$dS_u = \sigma dW_u$$

2. Market volatility is assumed to be constant

3. The density of the size of market order obeys a power law

$$f^Q(x) \propto x^{-1-\alpha}$$

where α is a hyperparameter approximately equal to 1.5 for major markets around the world.

4. Market order arrival follows a Poisson process with rates

$$\lambda(\delta) = Ae^{-k\delta}$$

where $\lambda^a(\delta^a)$ and $\lambda^b(\delta^b)$ represents the rate where the sell and buy orders will be matched respectively, with $\delta^a = p_t^a - s_t \; and \; \delta^b = s_t - p_t^b$ representing the distance between market mid price and ask/bid order price respectively.

5. Market has no interest rates

6. Limit orders can be continuously updated with no cost

These assumptions together model the order book dynamics for implementing the foundation of the reinforcement learning environment for the purpose of this study.

To maintain consistency with established literature, the parameter space selected for this study aligns with the simulation parameters outlined in the seminal Avellaneda and Stoikov paper where s = 100, T = 1, σ = 2, dt = 0.005, q = 0, γ = 0.1, k = 1.5 and A = 140.

4. Reinforcement Learning Settings

The following section defines the 3 key components for a reinforcement learning experiment – action space, state space and the reward function.

4.1. Action Space

The action space defines a set of all possible action for agent to interact with the environment. Depending on the algorithm, an action space can be discrete or continuous in nature. In particular, the continuous agent's action space is defined as a certain displacement to the current market mid price,

$$A_t = [\delta^b, \delta^a]$$

where the agent will then place bid and ask quotes on $p_b = s - \delta^b$ and $p_a = s + \delta^a$ respectively with $\delta^b \; and \; \delta^a$ both bounded between 0 and parameter *max_displacement d*, which is equal to 1.5 for the purpose of this study.

To discretize the continuous space, let

$$\delta^b, \delta^a \in \{\frac{1}{5}d, \frac{2}{5}d, \frac{3}{5}d, \frac{4}{5}d, d\}$$

similar to continuous action space, let the bid and ask quotes to be $p_b = s - \delta^b$ and $p_a = s + \delta^a$ respectively. Given that both $\delta^b \; and \; \delta^a$ are sampled from the same set of 5 elements, there will be a total of 25 possible combinations for the agent to place the quote pair. The hyperameter *d* is also taken to be 1.5 similar to the implementation of continuous space

4.2. State Space

The state space defines a set of all possible states that an agent can visit and quantify how the agent observe the current environment. In reinforcement learning, the state signal should retain all relevant information, that is the state signal should possess the Markov property, or should be a good basis for future prediction even if the environment is non-Markov in nature, such as the stock market.

While conventional studies often incorporate market variables such as order book imbalances and price changes in the state space, this study deviates by focusing solely on agent-related factors and omitting market variables (Carlsson & Regnell, 2022). This is based on the assumption of a consistent market behaviour modelled order book arrival rate and a market price driven by a constant volatility Brownian Motion. In particular, the state space is defined as,

$$s_t = (q_t, T - t),$$

where q_t denotes the market maker's inventory level and $T - t$ denotes the time till termination, represented by a normalized value between 0 and 1. Note a negative q_t represents a short position, vice versa. The absolute inventory level is also bounded within q_{max} units, that is, a discretized inventory level $q_t \in \{-\mathrm{q_{max}}, \dots, 1, 0, 1, \dots, \mathrm{q_{max}}\}$.

4.3. Reward Function

The reward function serves as a critical component that quantifies the immediate feedback an agent receives from its actions within an environment. It essentially defines the objective or goal of the agent's decision-making process and guides the agent's learning process.

This study compares the learning efficiency and simulation outcome of different reward functions from existing literature.

<u>*PnL with weighted inventory penalization*</u>

$$R_{t+1} = (s_{t+1} - p_b)\mathbb{1}\{dnb\} + (p_a - s_{t+1})\mathbb{1}\{dna\} - \lambda|q_{t+1}|$$

Full code implementation can be found at https://github.com/crxxom/RL-AS

where dnb(dna) is the number of bid(ask) orders placed at t=t that is and λ is a hyperparameter to control the extent of inventory penalization (Gasperov & Kostanjcar, 2021).

Asymmetrically dampened PnL

First denote,

$$\Psi_t = (s_{t+1} - p_b)\mathbb{1}\{dnb\} + (p_a - s_{t+1})\mathbb{1}\{dna\} + q_t(s_{t+1} - s_t)$$

as the non-dampened PnL.

The asymmetrically dampened PnL is therefore,

$$R_{t+1} = \Psi_t - max[0, \eta \cdot q_t (s_{t+1} - s_t)]$$

where η is the dampened factor, which is set to be 0.6 for the purpose of this study.

The idea of using the asymmetrically dampened PnL as the reward function is similar to the idea of adding an inventory penalization in the form of magnifying the punishment of wealth loss from holding inventory without magnifying the wealth gain symmetrically, discouraging the market maker from holding a large inventory (Spooner et. al., 2018).

Mean-variance formulation

$$R_{t+1} = \delta w_t - \frac{\kappa}{2}(\delta w_t - \hat{u})^2$$

where δw_t is the single tick return and $\hat{u}$ represents the rolling mean of δw. This mean-variance formulation of the utility function essentially guides the market maker to maximize wealth while minimizing the volatility of wealth change (Selser et. al., 2021).

5. Reinforcement Learning Algorithms

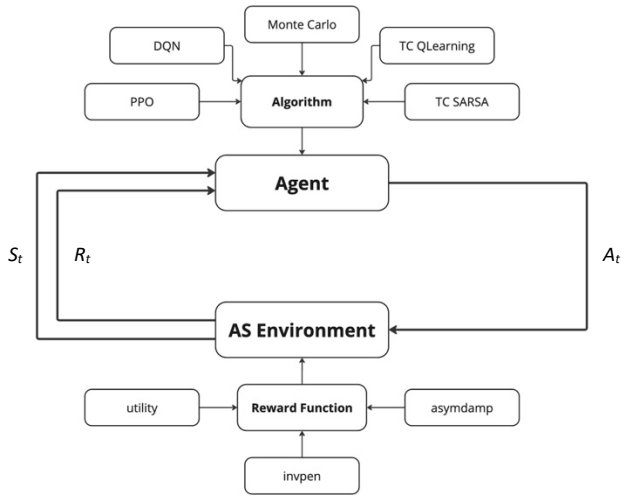
This study evalutes and compare the performance of different algorithms, including tabular learning methods and deep reinforcement learning methods. In particular, the following algorithms are trained and compared against the 2 benchmarks, the Avellaneda Stoikov agent and symmetric agent, on a number of evaluation metrics.

Algorithm	State space	Action space	Deep RL?
PPO	continuous	continuous	✓
DQN	continuous	discrete	✓
first-visit Monte Carlo	discrete	discrete	
Tile Coded Q Learning	discrete	discrete	
Tile Coded SARSA	discrete	discrete	

Full implementation of algorithms can be found in the github repository.

6. Experiment

6.1. Methods



The above diagram demonstrates the workflow of the experiments. A total of 15 base experiments will be conducted, taking all possible combinations among 5 algorithms and the 3 dedicated reward function.

The reinforcement agent will first go through the training stage in which it will interact with the environment for 2,000,000 steps, which is equivalent to 10,000 complete episodes. The trained model will then run through 5,000 monte carlo simulations on the same AS environment for evaluate the performances of the respective algorithm-reward function pair.

For the baseline experiments, the hyperparameters is taken as followed. *Discount factor* γ=0.97, *learning rate* α = 0.001, *epsilon* ϵ = 0.7, *epsilon decay factor*= 0.95, $max(|q|)$ = 20, price tick = 0.01, inventory penalty λ = 0.1, dampening factor η = 0.7.

6.2. Performance metrics

The following evaluation metrics are used to conduct quantitative analysis and comparison on the profit gaining effectively and risk management ability of the trained agents.

Terminal Wealth

The accumulated wealth at the end of episode denoted by W_T.

Wealth Volatility

$$\sigma_w = \sqrt{\frac{1}{T}\sum_{t=0}^{T}(W_t - \overline{W})^2}$$

which is represented by the standarad deviation of wealth over each simulated episode.

Episodic Return

$$G_t = \sum_{k=0}^{T} \gamma^k R_{t+k+1}$$

where we take $\gamma = 1$ as undiscounted return and the episodic return is denoted by G_0.

Mean Absolute Position (MAP)

$$MAP_t = \frac{1}{T}\sum_{t=1}^{T} q_t$$

which is a metric commonly used to account for the inventory risk.

Rolling PnL-to-MAP Ratio (PnLMAP)

$$PnLMAP_t = \frac{W_t}{MAP_t}$$

accounting for both profitability and inventory risk management of the agent (Gasperov & Kostanjcar, 2021).

6.3. Results

Table 1 (appendix) shows the complete simulation results sorted by PnLMAP, as well as the results of different test statistics to compare the statistical significance between experiments.

6.3.1. General results

In total, 3 experiments (PPO_invpen, DQN_invpen, PPO_asymdamp) demonstrated a better score in PnLMAP while 1 experiment (PPO_utility) demonstrated a better score in average terminal wealth than the optimum AS model under the AS environment. This is possible due to the fact that the derived equations in the AS model is also a functional approximation (Avellaneda & Stoikov, 2008).

The experiment results demonstrate the strength of reinforcement learning methods, especially deep reinforcement learning methods, in functional approximation to make optimal actions driven by their respective reward functions. At the same time, this also show the potential of the use of reinforcement learning in finance given that reinforcement learning only requires to learn through interaction with an environment, in which we have perfect understanding of, instead of requiring to solve for the environment dynamics which can be very unpredictable and difficult to solve for.

On the other hand, there are still several limitations yet to overcome for leveraging reinforcement learning in production. First, reinforcement agents typically requries a huge number of computational resources for the policy to converge during the training stage. At the same time, a greater problem is that it is difficult to understand the decision-making process of reinforcement learning agents, particularly deep reinforcement learning methods. This makes deploying related technology in production unrealiable considering that reinforcement learning is highly sensible to hyperparameters and unable to react with states that have never occuered in the training process, making it vulnerable to black swan or relevant events.

6.3.2. Comparison between reward functions

The simulation results demonstrate the importance of reward function in modelling the characteristic of the reinforcement learning agent.

This can be shown by comparing the MAP metric between agents that is guided with and without an inventory penalized reward function. In fact, all 4 agents with MAP lower than 1 leverages a reward function that directly (invpen) or indirectly (asymdamp) punish the accumulation of inventory, while agents that does not include inventory penalization (utility) tends to accumulate a higher average inventory when compared with other agents.

On the other hand, given that reward function with inventory penalization put a higher emphasis on inventory risk mangement, simulation results also demonstrate the characteristic of an average higher return given higher risk. In fact, PPO_utility gives the highest mean terminal wealth across 5000 simulations among other agents.

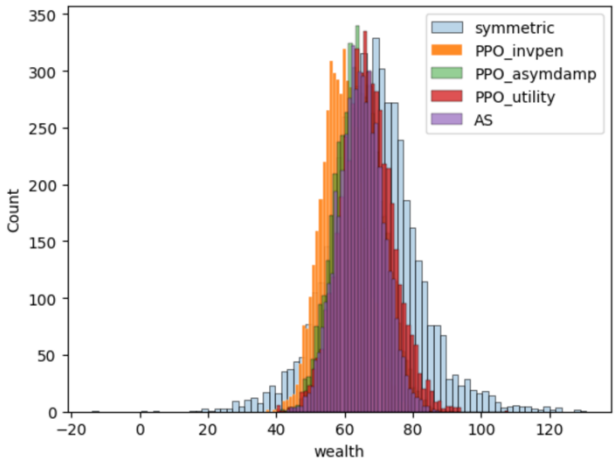


Figure1. Distribution of terminal wealth among 5000 simulations for algorithms of each reward function with highest PnLMAP

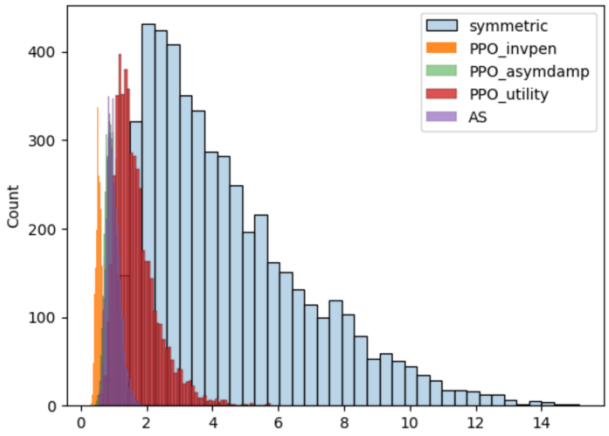


Figure2. Distribution of MAP among 5000 simulations for algorithms of each reward function with highest PnLMAP

6.3.3. Comparison between algorithms

In theory, all algorithms with a soft policy (non-deterministic) must converge given infinity amount of training data. So

given an equivalent environment, it will be more meaningful to compare the convergence rate of different algorithms, or the performance given same amount of interaction steps between the agent and the environment. The simulation results shows that in general, deep reinforcement learning demonstrate a stronger performance in both profit taking and risk managing when compared with tabular learning methods.

This may be the result of the high sparsity of tabular learning methods as mentioned in (Selser et. al., 2021), which shows that there are less than 1% of non-empty values in the q table after training. While on the other hand, deep reinforcement learning leverages the use of neural networks for functional approximation and requires much less experiences to train effectively.

The simulation results also show that algorithms with a continuous action space (PPO) have a stronger performance than algorithms with a discrete action space as according to figure 1 and figure 2, where PPO shows the highest average PnLMAP for all 3 types of reward functions when compared against other reward functions. This is inline with our expectation given that a continuous action space allows the agent to have a more precise quoting price when compared with algorithms with a discrete action space. In standard industrial practice, algorithms with discrete action space will be constructed based on the price tick of the asset, but for simplicity and training efficiency, one step is taken as 0.3 for the construction of the discrete action space instead of 0.01, the standard price tick for stocks in US market.

Another interesting observation is the training effectively of different algorithms. Figure 3. below illustrates the normalized episodic reward throughout the 10,000 training episodes. Similiar to the conclusion made above, deep reinforcement learning methods general demonstrate a faster and more stable convergence comparing to traditional tabular methods as evidenced from the lineplot in figure 3.

Moreover, a comparison between SARSA_asymdamp and TCSARSA_asymdamp demonstrate the impact of leveraging tile coding in discretizing the continuous state space on the training efficiency as shown by the difference in stability between the two algorithms over the training period.

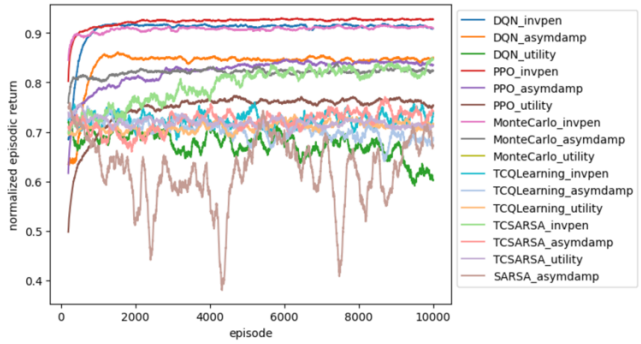


Figure3. Normalized episodic return over 10,000 training episode for the 16 repsective experiments

7. Conclusion

This study evaluates the performance of the use of reinforcement learning under a theoretical market making environment in terms of the agent's profit taking and inventory risk management capability. Simulation results show that reinforcement learning methods, particularly deep reinforcement methods is capable to give a better functional approximation that the optimum Avellaneda Stoikov model under the Avellaneda Stoikov market making environment. It also shows that deep reinforcement learning methods generally demonstrate a better performance in terms of the performance metrics on both profit taking and inventory management, as well as demonstrating a stronger training efficiency in terms of the convergence rate and training stability. At the same time, a comparison of experiements with different reward functions also demonstrate the role of reward function in the guidance of the agent learning process, showing that agents driven by a inventory penalized reward function demonstrate a stricter control in the mean absolute position over simulation episodes.

Despite the positive simulation results shown, a key limitation of this study lies on the unrealistic assumptions made in the modelling of the theoretical market making environment. This severly limits the consideration of market microstructure as well as incapability to capture the impact of queue position which plays a significant role especially in large tick assets.

Key areas for future research include the use of reinforcement learning simulated on historical limit orderbook data, the evaluation of the impact of adversial reinforcement learning as introduced in several existing literatures as well as simulating the experiments on different variations of the Avellaneda Stoikov environment.

Full code implementation can be found at https://github.com/crxxom/RL-AS

Appendix

Table 1. Simulation Results sorted by PnLMAP

	Terminal Wealth	Wealth std.	MAP	Episodic Return	PnLMAP
PPO_invpen	59.495140	17.280501	0.547150	48.623436	110.986514
DQN_invpen	58.079000	16.942983	0.762689	42.916880	77.647280
PPO_asymdamp	64.227526	18.680463	0.951159	56.280437	70.287214
Avellaneda Stoikov	64.726417	18.777720	0.989759	NaN	68.009254
DQN_asymdamp	63.494920	18.492532	0.972946	53.999249	67.535055
PPO_utility	66.595730	19.354344	1.691298	59.935688	44.490531
TCSARSA_invpen	62.967864	18.517524	2.493774	13.196360	29.940834
Symmetric	68.118599	20.044692	4.462171	NaN	20.521413
TCQLearning_invpen	60.293660	17.877204	4.289016	-25.073020	17.952887
TCQLearning_utility	63.527392	18.991725	5.159219	41.894760	15.834781
TCSARSA_asymdamp	59.945680	17.838038	5.400419	7.220974	15.478619
TCSARSA_utility	50.288384	15.082727	4.717490	29.420079	15.065617
MonteCarlo_utility	57.047312	16.863103	4.689593	37.622665	15.002305
TCQLearning_asymdamp	61.967388	18.563349	5.875580	4.697482	14.915942
MonteCarlo_asymdamp	57.625344	17.003658	4.933910	9.555188	14.174148
MonteCarlo_invpen	57.457892	17.063182	5.269777	34.422830	13.083604
DQN_utility	58.506812	18.039808	7.210589	20.119820	11.031349

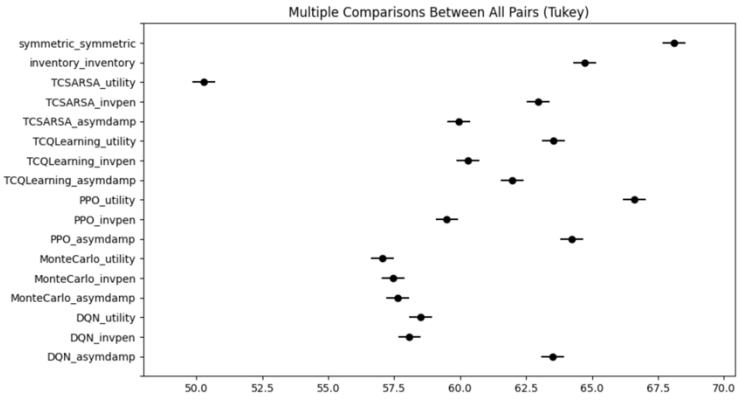


Fig4. Tukey's HSD test on terminal wealth

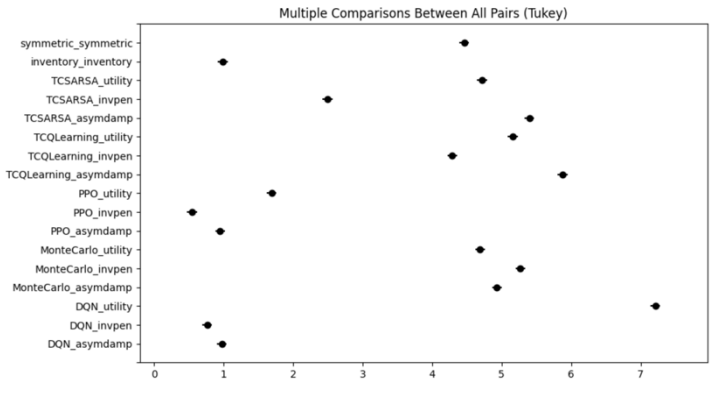


Fig5. Tukey's HSD test on MAP

Full code implementation can be found at https://github.com/crxxom/RL-AS

References

Selser, M., Kreiner, J., & Maurette, M. (2021). Optimal market making by Reinforcement Learning. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3829984

Avellaneda, M., & Stoikov, S. (2008). High-frequency trading in a limit order book. *Quantitative Finance*, *8*(3), 217–224. https://doi.org/10.1080/14697680701381228

Gasperov, B., & Kostanjcar, Z. (2021). Market making with signals through deep reinforcement learning. *IEEE Access*, *9*, 61611–61622. https://doi.org/10.1109/access.2021.3074782

Spooner, T., Fearnley, J., Savani, R., & Koukorinis, A. (2018). *Market making via Reinforcement Learning*. arXiv.org. https://arxiv.org/abs/1804.04216

Carlsson, S., & Regnell, A. (2022). *Reinforcement learning for market making*. DIVA. https://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A1695877&dswid=8770

Xu, Z., Cheng, X., He, Y., Ziyi XuPeking University, P., Xue ChengPeking University, P., & Yangbo HePeking University, P. (2022, May 9). *Performance of deep reinforcement learning for high frequency market making on actual tick data: Proceedings of the 21st International Conference on Autonomous Agents and multiagent systems*. ACM Conferences. https://dl.acm.org/doi/10.5555/3535850.3536103

Hambly, B., Yang, H., & Xu, R. (2023). Recent advances in reinforcement learning in Finance. https://onlinelibrary.wiley.com/doi/full/10.1111/mafi.12382

Richard, S. & Andrew G. (2014). Reinforcement Learning: An Introduction. The MIT Press