Applications of Reinforcement Learning in Automated Market-Making

Mohammad Mani
Department of Informatics
King's College London
mohammad.mani@kcl.ac.uk

Steve Phelps
School of Computer Science
University of Birmingham
sphelps@sphelps.net

Simon Parsons
Department of Informatics
King's College London
simon.parsons@kcl.ac.uk

ABSTRACT

A key problem for the design and analysis of trading agents that are deployed in financial markets is coping with risk-sensitivity, but this problem has received relatively low attention from the AI community. In this paper, we introduce an approach for automated and intelligent market-making using risk-sensitive reinforcement learning. The market-maker has multiple objectives, including but not limited to making profit, controlling inventory and controlling market quality. We model the agent-environment interaction by a Markov decision process. We simulate the financial market using the Glosten-Milgrom information model. We show that our automated market-making agent can intelligently learn to achieve its objectives by interacting with other trading agents in the simulated environment. As expected, we find that in most cases, risk-averse market-makers who follow deterministic policies tend to make less profit compared to risk-neutral market-makers. However, when we use a Boltzmann softmax action-selection rule, the resulting policy yields more profit while also maintaining a low level of inventory and an acceptable level of market quality.

KEYWORDS

Reinforcement Learning; Learning and Adaptation; Agent-based Simulation; Financial Market

1 INTRODUCTION

A majority of trading activities in equities markets occur through financial exchanges. There are two main types of orders submitted to such exchanges: limit orders and market orders. Limit orders refer to orders with specified limit price and quantity. The limit price is the maximum (minimum) price the buyer (seller) would be willing to pay for the specified quantity. Market orders on the other hand refer to orders that only specify a quantity to be traded at the best available bid or ask price. Market orders guarantee execution, whereas there is no guarantee on limit orders being fully executed.

In addition, there are two main types of market mechanisms deployed in modern financial exchanges: order-driven and quote-driven. Order-driven mechanisms refer to the scenario where buyers and sellers submit their limit or market orders to the exchange and the orders are processed by a price-time priority matching engine. This mechanism implements a double sided auction. The unmatched limit orders are collected into the limit order book. The matching mechanism can occur continuously, where the order book is updated after the insertion of an order or discretely, where the order-book is updated in a pre-specified intervals. Market participants in an order-driven market may implement mixed trading strategies including market-making. The quote-driven mechanism on the other hand refers to the scenario where the exchange designates a specific market-maker that must offer both competitive buy and sell prices to market participants. The quote-driven market reduces transparency for market-participants, but provides a greater degree of liquidity to the market.

Market-making refers to a class of trading strategies where a market participant submits both buy and sell limit orders to the exchange. The main role of market-makers is to provide liquidity. More specifically, a market-maker quotes competitive limit orders to buy and sell on both sides of the limit order book in order to make a profit from the bid-ask spread while also providing liquidity to other market participants. In general, market-makers are exposed to two main sources of risk [14]: 1) the risk of holding inventory; the market-maker aims to hold zero inventory by a given trading horizon and 2) The risk of adverse selection; the market-maker may face traders with better information about the true or the fundamental value of the asset. The market-maker has therefore multiple objectives including, but not limited to, making a profit, controlling its inventory, and controlling its quoted bid-ask spread, all over a pre-specified time-horizon and in a competitive environment.

In this paper, we introduce a framework for automated market-making using risk-sensitive reinforcement learning. We model the agent-environment interaction by a finite Markov decision process in discrete-time with a linear reward function. We simulate the financial market using the Glosten-Milgrom[1] information model [10]. Our market-making framework is an extension to the Chan-Shelton market-making model, studied extensively in [6], [7] and [17]. Our risk-sensitive reinforcement learning framework is based on Mihatsch-Neuneier one-step temporal difference learning algorithms [13]. While all of these components have been studied before, our approach provides a novel combination, in particular the use of risk-sensitive learning as a way to develop a market-making strategy.

In this paper, we present experimental results for both the risk-neutral (standard) and the risk-sensitive versions of on-policy SARSA and Double-SARSA reinforcement learning algorithms. The SARSA algorithm is introduced in [16] and is examined further in [20] and [19]. The numerical analysis of reinforcement learning algorithms with double learning is also studied extensively in [21].

GAIW: Games, Agents and Incentives Workshops, May 2019, Montreal, Canada

https://doi.org/doi

[1]The Glosten-Milgrom model is used as a baseline for initial investigation. More advanced order-driven market models will be investigated in the future work.

We show that our risk-sensitive market-making agent can learn to achieve its objectives by interacting with other trading agents in the simulated environment. We find that in most cases risk-averse market-makers who follow deterministic policies tend to make less profit compared to risk-neutral market-makers. However, when we use a Boltzmann softmax action-selection rule, the resulting policy yields more profit than the standard approach while also maintaining a low level of inventory and an acceptable level of market quality. By market quality, we mainly refer to the bid-ask spread of the market-making agent. Lower spread generates higher market quality but lower profit for the market-maker, whereas higher spread generates more profit for the market-maker and lower market quality for other trading agents. It is worth mentioning that competition between market-makers naturally results in lower market spread. In this paper, however, we focus on a single market-maker with bounded bid-ask spread.

2 RELATED WORK

The Glosten-Milgrom information model [10] describes the micro-structure price formation process and explains the existence of a bid-ask spread as a consequence of a market-maker's adverse selection. In this model, a single market-making agent interacts directly with a group of informed and a group of uninformed trading agents in order to trade a unit share of an asset over a pre-specified time-horizon $[0, T]$. In addition, it is assumed that a stochastic process, denoted by $(P_t)_{t\in[0,T]}$ exists that shows the evolution of the true or the fundamental value of the asset. The fundamental value is assumed to be only affected by exogenous factors. Moreover, it is assumed that the market-maker and the uninformed traders have only access to publicly available information, whereas, the informed traders have access to superior private information about the fundamental value of the asset. In particular, informed traders know the exact realisation of $(P_t)_{t\in[0,T]}$.

Chan-Shelton [7] implemented the Glosten-Milgrom information model and used a standard (risk-neutral) on-policy SARSA reinforcement learning algorithm with epsilon-greedy action-selection rule as well as Monte Carlo simulation to track the fundamental value $(P_t)_{t\in[0,T]}$. In terms of market-making, tracking this stochastic process refers to quoting ask and bid prices such that they enclose and bound the fundamental value. In this case, the market-maker only trades with the uninformed traders and consequently makes a profit.

A similar market-making framework is studied in [4], [8] and [9]. They used Bayesian learning as opposed to reinforcement learning in order to learn a prior over the likelihood of the true or the fundamental value of the asset. In this study, we ignore the explicit estimation of this stochastic process and instead rely on the insights gained from the properties of the Markov decision process to track this stochastic process.

The literature on theoretical and experimental market-making models is extensive, however, very few of those used the Glosten-Milgrom information model or machine learning techniques to investigate optimal strategies. We therefore focus our attention to mainly extend the Chan-Shelton [7] and Das [8] market-making models.

3 SIMULATION OF FINANCIAL MARKET

We implement the Glosten-Milgrom information model [10] with a single market-making agent, a group of informed and a group of uninformed trading agents. The target trading volume is a unit share of the asset. We also simulate an exogenous integer-valued stochastic process $(P_t)_{t\in[0,T]}$ that describes the evolution of fundamental value of the asset.

In our simulation, the market-making agent submits only passive limit orders whereas other trading agents (informed or uninformed) can only submit aggressive market-orders. This assumption is made in order to guarantee full execution of orders submitted by informed or uninformed agents.

Uninformed traders are equally likely to buy or sell a unit share of the asset at any given time. Informed traders on the other hand, implement a buy-low and sell-high strategy. It is assumed that informed and uninformed traders are not allowed to interact with each other and can only trade with the market-maker. The market-maker's quotes are guaranteed to be executed by a trader. It is not known to the market-maker which type of trader executed the order at the ask or at the bid price, and the market-maker has no prior knowledge about the fundamental value $(P_t)_{t\in[0,T]}$. The market-making agent simply has to learn and adapt to these events in order to find an optimal and profitable strategy.

In addition, there is a guaranteed arrival of an event at each time-step. There are five events in total including the arrival of an informed trader with probability λ_i, the arrival of an uninformed buyer with probability λ_u, the arrival of an uninformed seller with probability λ_u as well as upward jump in the fundamental value with probability λ_p and downward jump in the fundamental value with probability λ_p. Thus the relation $2\lambda_p + 2\lambda_u + \lambda_i = 1$ holds at all time-steps.

4 ALGORITHMIC MARKET-MAKING

We treat the market environment as a Markov decision process which the market-maker solves using reinforcement learning.

4.1 Markov Decision Process (MDP)

We model the sequential decision making task of the market-maker by a MDP. A finite, stationary MDP is a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P} \rangle$, where, $\mathcal{S}$ is a finite collection of states, $\mathcal{A}$ is a collection of finite sets of possible actions, $\mathcal{R}$ is the set of possible values of the immediate rewards and $\mathcal{P}$ describes the transition probabilities. In particular, $\mathcal{P}(s', r \mid s, a) = \mathbb{P}(S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a)$ for $s, s' \in \mathcal{S}$, $a \in \mathcal{A}$ and $r \in \mathcal{R}$, where $\mathbb{P}$ is a probability measure over the sample space Ω.

In this model, $S_t = (IMB_t, SP_t) \in \mathcal{S} \subseteq \mathbb{Z} \times \mathbb{N}$ is the set of states at time t, $A_t = (\Delta ASK_t, \Delta BID_t) \in \mathcal{A} \subseteq \mathbb{Z} \times \mathbb{Z}$ is the set of actions at time t and $R_t = w_{pro}\Delta PRO_t - w_{inv}|INV_t| - w_{spr}SP_t$ is the immediate reward at time t.

The immediate reward is a linear function of factors with different weight parameters w_{pro}, w_{inv} and w_{spr} for profit, inventory and market-maker's bid-ask spread. IMB_t denotes the volume imbalance. The volume imbalance is determined as the number of buy orders minus the number of sell orders for a unit share of the asset. $SP_t = ASK_t - BID_t$ denotes the market-maker's bid-ask spread. The spread is non-negative since $ASK_t \geq BID_t$. INV_t denotes the market-maker's inventory and $\Delta ASK_t = ASK_t - ASK_{t-1} \in \mathbb{Z}$ and $\Delta BID_t = BID_t - BID_{t-1} \in \mathbb{Z}$ correspond to the changes in the market-maker's quotes. The market-maker's profit at every time-step is computed as follows.

$$\Delta PRO_{t+1} = \begin{cases} ASK_t - P_t & \text{If a trader buys} \\ P_t - BID_t & \text{If a trader sells} \end{cases}$$

The agent-environment dynamics is as follows: at time t, the market-maker observes the state $S_t = s$ and takes action $A_t = a$ according to a stochastic, stationary and Markovian policy. The action of the market-maker updates the bid and ask prices from the previous time-step and creates a new pair (ASK_t, BID_t). The traders then observe this change and react to this price based on their level of information. In particular, informed traders enter the market with probability λ_i, and observe P_t and (ASK_t, BID_t) and submit market order buy if $ASK_t < P_t$, market order sell if $BID_t > P_t$, and nothing at all if $BID_t \leq P_t \leq ASK_t$. Uninformed traders, on the other hand, only observe (ASK_t, BID_t) and submit market-order buy or sell with equal probability λ_u. At time $t+1$, transition to the new state $S_{t+1} = s'$ occurs and an immediate reward $R_{t+1} = r$ is received by the agent. The new state and the immediate reward are jointly distributed according to the transition probability $\mathcal{P}(s', r \mid s, a)$. The agent then observes S_{t+1}, takes action A_{t+1} and traders react to the adjusted quotes. The dynamics then continues until the end of the episode at time $t = T$. The agent-environment interaction is depicted in Figure (1).

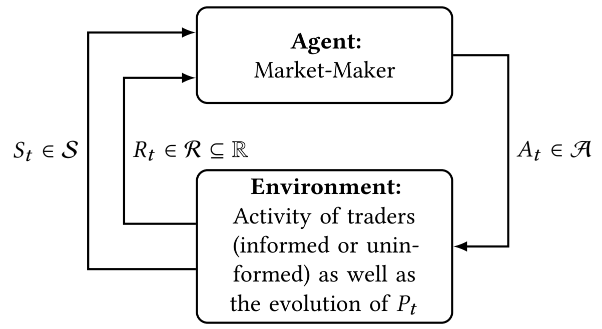

Agent:
Market-Maker
$S_t \in \mathcal{S}$
$R_t \in \mathcal{R} \subseteq \mathbb{R}$
$A_t \in \mathcal{A}$
Environment:
Activity of traders (informed or uninformed) as well as the evolution of P_t

Figure 1: The agent environment interaction in a MDP with state S_t, action A_t and reward R_t at time t. P_t is a stochastic process that shows the evolution of the fundamental value of the asset

4.2 Reinforcement Learning (RL)

The main goal of the market-making agent is to find the optimal policy that updates its bid and ask prices dynamically such that long term expected profit is maximised. In addition, the market-maker also needs to control its inventory as well as its quoted bid-ask spread. In the context of MDPs, the agent aims to maximise the expectation of the sum of discounted immediate rewards or the return. The agent's action-selection process is identified by a policy π. For simplicity, we consider only stationary policies, therefore π is time-invariant. More specifically, action A_t is selected according to $\pi(. \mid S_t)$. A time-invariant, Markovian and stochastic policy is a mapping $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ such that $\pi(a \mid s) = \mathbb{P}(A_t = a \mid S_t = s)$. We use stochastic policies to ensure sufficient exploration by the learning agent.

For a discount factor $\gamma \in (0, 1)$, the episodic return is defined by $\sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1}$. The state-action value function, $Q_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, for a fixed policy π is given by

$$\begin{aligned} Q_\pi(s, a) &\doteq \mathbb{E}_\pi[\sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma \sum_{a'} Q_\pi(S_{t+1}, a')\pi(a' \mid S_{t+1}) \mid S_t = s, A_t = a] \end{aligned} \tag{1}$$

The optimal state-action value function is therefore given by

$$\begin{aligned} Q_*(s, a) &\doteq \max_\pi Q_\pi(s, a) \\ &= \mathbb{E}[R_{t+1} + \gamma \max_{a'} Q_*(S_{t+1}, a') \mid S_t = s, A_t = a] \end{aligned} \tag{2}$$

The optimal policy is then obtained by $\pi^* = \arg\max_a Q_*(s, a)$. Moreover, the transition probabilities, $\mathcal{P}(s', a \mid s, a)$ are not known explicitly, therefore we need to rely on approximation techniques to solve for the optimal policy. To do this we use temporal difference methods. Unlike pure dynamic programming, temporal difference learning algorithms do not require a complete model of the environment. In addition, they can take advantage of the structure in MDPs. This makes temporal difference learning a good candidate for the reinforcement learning market-maker.

The SARSA algorithm with a learning rate $\alpha \in \mathbb{R}$, is a simple iterative on-policy temporal difference learning algorithm that is proved to converge to optimality [18]. The update rule for SARSA is given by

$$\begin{aligned} Q(S_t, A_t) &\leftarrow Q(S_t, A_t) \\ &+ \alpha(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)) \end{aligned} \tag{3}$$

A natural extension to the SARSA algorithm is the Double SARSA algorithm. It is shown that double-learning improves the standard single-learning by eliminating the maximum bias caused by estimating the Q-values. Maximisation bias and double learning are introduced and extensively investigated in [23] and [24]. In essence, the agent learns two independent Q-value estimates. It uses one for the selection of the best action and the other to update Q-values for the action selected. This reduces the chances of both estimators overestimating the same action. The Double-SARSA algorithm has two update rules:

$$\begin{aligned} Q^{(1)}(S_t, A_t) &\leftarrow Q^{(1)}(S_t, A_t) \\ &+ \alpha(R_{t+1} + \gamma Q^{(2)}(S_{t+1}, A_{t+1}) - Q^{(1)}(S_t, A_t)) \end{aligned} \tag{4}$$

$$\begin{aligned} Q^{(2)}(S_t, A_t) &\leftarrow Q^{(2)}(S_t, A_t) \\ &+ \alpha(R_{t+1} + \gamma Q^{(1)}(S_{t+1}, A_{t+1}) - Q^{(2)}(S_t, A_t)) \end{aligned} \tag{5}$$

The agent then picks each of (4) and (5) with probability 0.5. Both standard SARSA and standard Double-SARSA algorithms maximise the risk-neutral objective function. A risk-neutral agent is only concerned with maximising the expectation of the return. Such a risk-neutral agent is not concerned with the variance of the sum of discounted rewards when it comes to maximising the return.

However, in many real-world optimisation problems, in particular in finance, it is crucial to take into account some degree of risk-aversion. Doing so will tend to reduce bid-ask spread and the amount of inventory held.

4.3 Risk-sensitive RL

To take variance reduction into account, we developed a modified version of the the Mihatsch-Neuneier risk-sensitive reinforcement learning model [13]. The modification is to take into account the effects of double learning. The Mihatsch-Neuneier model is a simple and mathematically tractable framework for risk-sensitive reinforcement learning. The model is easily adaptable to one-step temporal difference learning algorithms. In Mihatsch-Neuneier model, the temporal difference errors are transformed during learning in every time-step instead of the return of the process.

Formally, let $f^{(\kappa)} : \mathbb{R} \rightarrow \mathbb{R}$ be a transformation function defined by

$$f^{(\kappa)}(x) = \begin{cases} (1-\kappa)\,x, & \text{If } x > 0 \\ (1+\kappa)\,x, & \text{Otherwise} \end{cases} \tag{6}$$

The function's argument x denotes the temporal difference error and $\kappa \in (-1, 1)$ controls the degree of risk sensitivity. The objective function is risk-avoiding if $\kappa > 0$ and risk-seeking if $\kappa < 0$. In addition, the objective function reduces to the risk-neutral case when $\kappa = 0$. The transformation function, appropriately weights positive and negative temporal difference errors during learning. It overweights transitions to successor states when the immediate rewards are smaller than in the average and it underweights transitions to states that promise a higher return than in the average. The risk-sensitive update rule for the SARSA algorithm is therefore given by:

$$\begin{aligned} Q(S_t, A_t) &\leftarrow Q(S_t, A_t) \\ &\quad + \alpha f^{(\kappa)}(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)) \end{aligned} \tag{7}$$

In addition, we have a risk-sensitive Double-SARSA algorithm which picks each of (8) and (9) with probability 0.5.

$$\begin{aligned} Q^{(1)}(S_t, A_t) &\leftarrow Q^{(1)}(S_t, A_t) \\ &\quad + \alpha f^{(\kappa)}(R_{t+1} + \gamma Q^{(2)}(S_{t+1}, A_{t+1}) - Q^{(1)}(S_t, A_t)) \end{aligned} \tag{8}$$

$$\begin{aligned} Q^{(2)}(S_t, A_t) &\leftarrow Q^{(2)}(S_t, A_t) \\ &\quad + \alpha f^{(\kappa)}(R_{t+1} + \gamma Q^{(1)}(S_{t+1}, A_{t+1}) - Q^{(2)}(S_t, A_t)) \end{aligned} \tag{9}$$

Balancing exploration and exploitation is an important issue in approximating Q-values. In order to guarantee sufficient exploration, the learning agent needs to explore random states that are not necessarily recommended by the greedy Q-values. A policy which is greedy with respect to Q-values is a deterministic policy. It is usually denoted only by a mapping $\pi : \mathcal{S} \rightarrow \mathcal{A}$. The agent that follows the greedy policy will pick the action $a \in \mathcal{A}$ when in state $s \in \mathcal{S}$ according to $a = \pi(s) = \arg\max_a Q(s, a)$. The ϵ-greedy, with parameter $\epsilon \in (0, 1)$, on the other hand is a stochastic policy $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ used extensively throughout reinforcement learning. The agent that follows an ϵ-greedy policy will select an action $a \in \mathcal{A}$ when in state $s \in \mathcal{S}$ according to the following probability distribution.

$$\pi(a \mid s) = \begin{cases} 1-\epsilon & \text{If } a = \arg\max_a Q(s, a) \\ \epsilon \,/ (\mid \mathcal{A} \mid -1) & \text{Otherwise} \end{cases} \tag{10}$$

The Boltzmann softmax policy [5] is a stochastic action-selection system that selects an action according to the Boltzmann probability distribution with parameter τ. The Boltzmann distribution is given by

$$\pi(a \mid s) = \mathbb{P}(A_t = a \mid S_t = s) = \frac{e^{Q(s,a)/\tau}}{\sum_{a' \in \mathcal{A}} e^{Q(s,a')/\tau}} \tag{11}$$

The Boltzmann softmax action-selection rule picks random actions while adjusting the policy more towards the greedy policy. This, however, depends on the choice of the parameter τ. For example, higher τ, results in more actions being selected according to a uniform distribution, whereas, lower τ, results in more actions being selected towards the greedy action.

Table 1: Parameters for the RL algorithms

Parameters	Value	Description
γ	0.9	Discount factor
α	0.004	Learning rate
ϵ	0.1	ϵ-greedy parameter
τ	0.75	Softmax parameter
κ	0.8	Risk-aversion parameter

Table 2: Parameters for the immediate reward function

Parameters	Value	Description
w_{inv}	0.1	Inventory weight
w_{spr}	0.1	Spread weight
w_{pro}	0.8	Profit weight

Table 3: Probabilities

Parameters	Value	Description
λ_u	0.132	Arrival of uninformed trader
λ_p	0.105	Jump up / down in the fundamental value
λ_i	0.526	Arrival of informed trader

5 EXPERIMENTS

5.1 Experimental Setup

We ran experiments in which we solved the MDP for the market-making agent using the following four algorithms:

A1: risk-neutral (standard) $\epsilon-$greedy SARSA given in update rule (3) with action selection from equation (10);
This is exactly the model used in [7].

A2: risk-sensitive $\epsilon-$greedy SARSA given in update rule (7) with action selection from equation (11);

A3: risk-sensitive ϵ-greedy Double-SARSA given in update rules (8) and (9) with action selection from equation (10); and

A4: risk-sensitive softmax Double-SARSA given in update rules (8) and (9) with action selection from equation (11).

In all cases we used a linear immediate reward function:

$$R = w_{pro}\Delta PRO - w_{inv}|INV| - w_{spr}SP$$

combining profit, inventory and market-maker's spread at every time-step. The aim of these experiments was to establish whether a market-maker can use these variants of reinforcement learning approaches to learn a policy for generating bids and asks that bound the (time-varying) fundamental value of the good being traded, and to establish how the risk-sensitive variants, A2–A4, perform in comparison with the standard, risk neutral, approach, A1 studied in [7].

Tables 1–3 show the parameters used for these experiments. Table 1 shows the parameters corresponding to the reinforcement learning algorithms with given in update rules (3)– (5) and (7)– (9). These are the values used in [7]. Table 2 shows the weight parameters corresponding to the linear immediate reward function and Table 3 shows the arrival probabilities of trading agents as well as the probability of up or down movements in the fundamental value. The values in Table 3 are also taken from [7]. The fundamental value $(P_t)_{t\in[0,T]}$ is simulated by a Poisson jump process. More specifically, the jump sizes are distributed according to a uniform distribution, hence, P_t moves either up or down with equal probability λ_p.

At the beginning of the simulation, the initial fundamental value of the asset is known to all traders as well as the market-maker. The initial fundamental value is set as $P_0 = 100$, and it is therefore natural for the market-maker to initialise its ask and bid quotes by $(101, 99)$. In addition, there is a numerical bound imposed on the set of states and actions. In particular, we used $IMB_t \in \{-1, 0, 1\}$, $SP_t \in \{1, 2, 3, 4\}$, $\Delta ASK_t \in \{-1, 0, 1\}$ and $\Delta BID_t \in \{-1, 0, 1\}$. Therefore, given an action-set A_t and a state-set S_t, there are 9 possible actions in every 12 possible states. Moreover, the simulation runs for 2000 learning sessions, each learning session runs for 4000 episodes and each episode consists of 250 time-steps.

The performance of the market-maker is measured by the following factors:

Average absolute price deviation

$$\overline{\Delta P} = \sum_{t=1}^{T} \mid BID_t - P_t \mid + \mid ASK_t - P_t \mid \tag{12}$$

Average absolute price deviation ΔP assesses the convergence of the reinforcement learning algorithms to the optimal policy. More specifically, it measures the distance between the true fundamental value P_t and the quotes of the market-maker (ASK_t, BID_t) for an episode. ΔP is minimised when the fundamental value is bounded by the quotes of the market maker (i.e. when $BID \le P_t \le ASK_t$).

End of episode inventory

End of episode inventory INV_T refers to the market-maker's inventory at the end of a trading horizon. It measures whether the agent liquidated or acquired all of its target positions successfully throughout the episode. The optimal market-making agent aims to achieve an end of episode target inventory close to zero.

End of episode profit

$$PRO_T = \sum_{t=1}^{T} \Delta PRO_t \tag{13}$$

End of episode profit PRO_T measures the profit made or the loss incurred at the end of an episode.

Average episodic spread

$$\overline{SP} = \frac{1}{T}\sum_{t}^{T} SP_t \tag{14}$$

Average episodic spread SP measures the average quoted spread for an episode.

5.2 Results

Figures 3a–3d show the average absolute price deviation for the four reinforcement algorithms A1–A4. This is a good metric to examine for convergence — when the agent can, as we want it to, accurately predict the fundamental price, its offers can bound the fundamental price and the absolute price deviation will be close to zero. The figures clearly show convergence, and further show that the slowest algorithm to converge is risk-sensitive Double-SARSA with softmax, A4, which converges to an optimal strategy after around episode 1500. The results for risk-neutral SARSA, A1, replicate the results of [7], and we would expect that softmax, A4, would take longer to converge because softmax gives more weight to exploration early in the learning process. In addition, we would expect single learning algorithms to converge faster than their equivalent double learning algorithms, and we see this in the performance of A2 and A3.

Another view on the process of convergence is given by Figure 2. This shows results from a series of learning episodes of the risk-sensitive Double SARSA algorithm with softmax (A4). The episodes in the top row of Figure 2 show the fundamental price of the asset (black line) and the market-maker's spread (grey shaded area). It is evident that, in the later episodes, the market-making agent tracks the fundamental value more accurately. The optimal region, corresponds to the region in which the algorithm has been trained to produce profitable strategies. In essence, the optimal region corresponds to a market-making agent that has learnt to successfully track and bound the fundamental value. Figure 2 also shows the way that the agent's performance in terms of profit and inventory differ between the optimal and non-optimal regions. Profit, shown in the middle row of Figure 2, decreases over the episodes in the non-optimal region, and increases over the episodes in the optimal region. Inventory, shown in the bottom row of Figure 2, is a large negative quantity (indicating that inventory has fallen a long way below target) in the non-optimal region while it is small amount above target in the optimal region.

Figures 3m–3t show the end of episode profit for A1–A4. Figures 3m–3p give a representation of the distribution of profits over the last 1000 episodes, and Figures 3q–3t give box plots for the average profit over the last episodes. This is non-negative for all

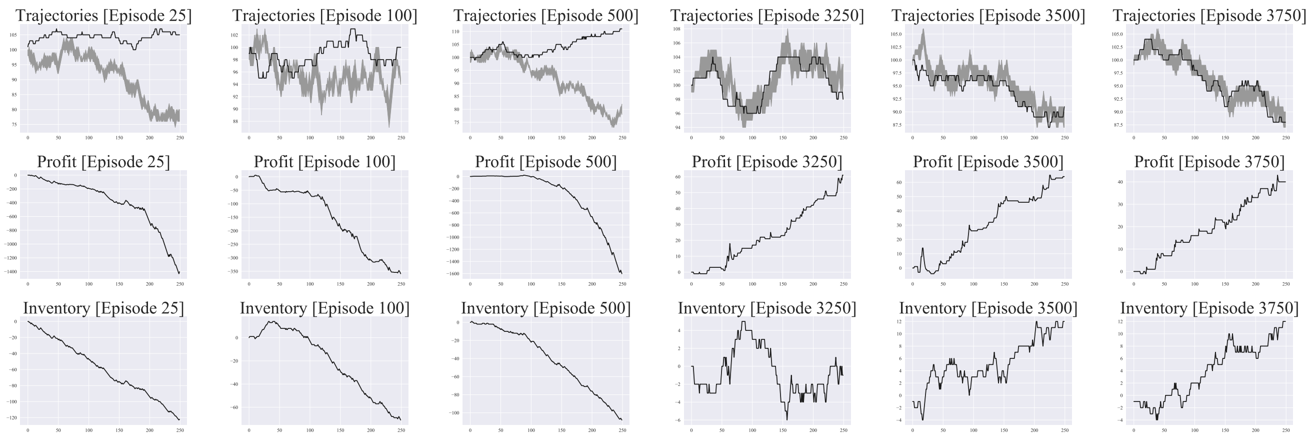


Figure 2: The evolution over time of metrics for risk-sensitive Double SARSA with softmax, A4. These include episodes from early in the learning process (the non-optimal region, episodes 25, 100 and 500) and from after learning has converged (the optimal region, episodes 3250, 3500, 3750). The graphs plot (top to bottom) bid/ask against true fundamental value, profit, and inventory. In the non-optimal region, inventory becomes negative.

algorithms, and and relatively high for the risk-sensitive agent using Double-SARSA and softmax. Figures 3e–3l show the inventory level in the optimal region, again both as a distribution, Figures 3e–3h, and as box plots Figures 3i–3l. The box plots, in particular, make it clear that all of A1–A4 have an optimal region inventory that is close to zero. It is worth noticing that, although inventory is not a state variable in the MDP, it is still minimised, and minimising inventory is desirable behaviour for the market-making agent. Finally, Figures 3u–3x shows the average episodic spread for all episodes including the optimal region.

5.3 Analysis

Figures 4a, 4b and 4c provide a head-to-head comparison of the four reinforcement learning algorithms, A1–A4. Figure 4a compares inventory, Figure 4b compares profit, and Figure 4c compares spread. The comparisons are made over the last 1000 episodes of the optimal region. To establish whether the differences in the values seen in Figures 4a, 4b and 4c are significantly different, we performed pairwise tests on all pairs of values shown in each figure using Welch's unequal variances t-test. The result is that all the differences are statistically significant ($p = 0.001$).

These results are consistent with our expectation for the risk-sensitive scenarios with ϵ-greedy, A2 and A3, since we obtain lower average profit, lower average spread and higher average inventory while also reducing the variance for all three factors as compared to the standard scenario, A1. However, interestingly, we find that the risk-sensitive softmax Double-SARSA, A4, not only results in reduced variance for the three factors in comparison to A1, as we would expect, but also yields what are arguably better results than A1, namely lower inventory, higher profit and lower spread. We consider that this shows A4 is better than A1 since a higher profit is clearly superior, while lower inventory means less capital tied up, and a large spread is only desirable in that it creates the opportunity for a larger profit it is not desirable per se. This result was, naturally, unexpected, since risk-sensitive frameworks, in general, produce less profit than risk-neutral frameworks. The result we see in this case could be the consequence of the risk-sensitive agent's more efficient exploration under the Boltzmann softmax policy. In any case, we conclude that, the use of softmax gives the agent more control over the fine tuning of the trade-off between risk and profit.

The performance that we see from the risk-sensitive Double-SARSA with softmax is what an optimal market-making strategy seeks to achieve — low variance and therefore low risk, high profit, low inventory and high spread conditional on the fact that the spread should balance itself with respect to the probability of execution. It is worth mentioning that, a market-maker with large spread has the potential to make more profit, but because the probability of their orders begin executed becomes smaller, the strategy may result in large inventory, which is suboptimal. Hence a balance between factors embedded in the reward function must be kept at all time to achieve an optimal trading strategy.

Finally, we note that the risk-sensitive framework has the same time and memory complexity as the standard risk-neutral framework. The immediate reward function is transformed to penalise the running inventories. This, reduces the dimension in the MDP state-space and also forces the agent to be more risk-averse. A stochastic policy is implemented directly via the Boltzmann softmax policy. This allows the agent to explore more and achieve higher average expected profit in the risk-sensitive case.

6 CONCLUSIONS & FUTURE WORK

In this paper, we introduced a framework for automated market-making using risk-sensitive reinforcement learning. We compared the performance of this risk-sensitive market-making agent with the performance of agents using various risk-neutral (standard) and risk-sensitive reinforcement learning algorithms using different types of action selection. In the context of market-making, we observed that risk-sensitive reinforcement learning with ϵ-greedy action selection results in lower average profit, lower average spread

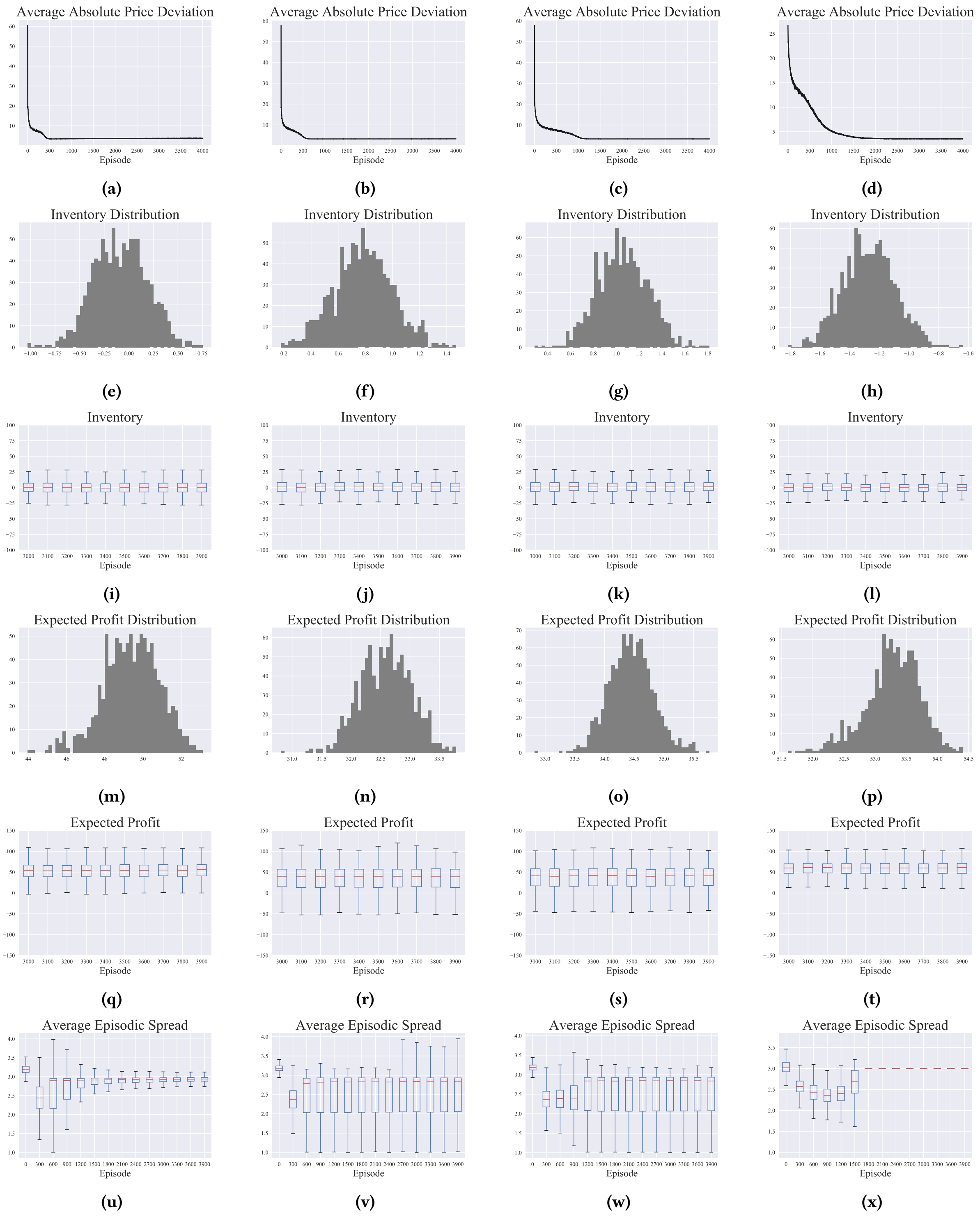
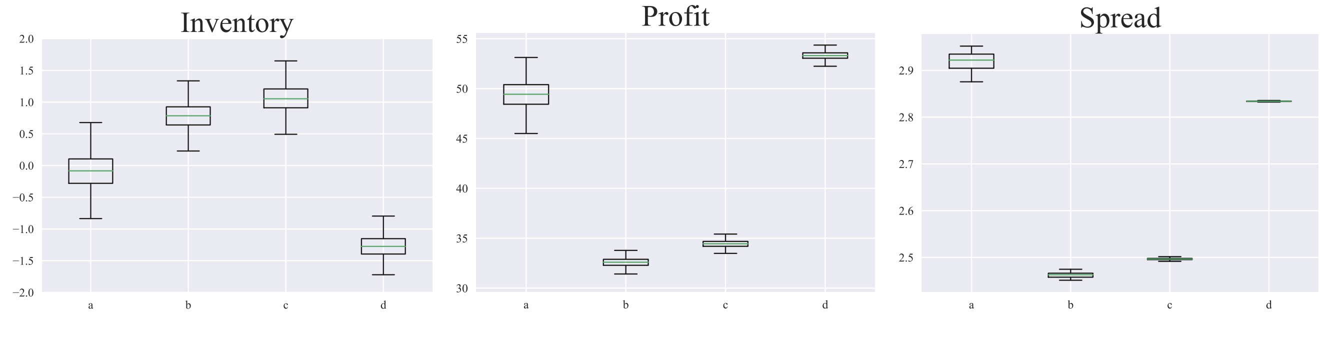


Figure 3: Results for risk neutral ϵ-greedy SARSA (a, e, i, m, q and u); risk sensitive ϵ-greedy SARSA (b, f, j, n, r and v); risk sensitive ϵ-greedy Double-SARSA (c, g, k, o, s and w); and risk sensitive softmax Double-SARSA (d, h, l, p, t, x). Note that in (x), after convergence the variance in the spread is so narrow that it is not visible.



(a) **(b)** **(c)**

Figure 4: Comparison of (a) inventory, (b) profit and (c) spread at the end of episodes in the optimal region across the four different RL algorithms. Within each plot, the algorithms are, left to right, Risk-neutral ϵ-greedy SARSA (A1); Risk-sensitive ϵ-greedy SARSA (A2); Risk-sensitive ϵ-greedy Double SARSA (A3); and Risk-sensitive Double SARSA Softmax (A4).

and lower average inventory as compared to the standard reinforcement learning approach, but has the benefit of reducing the variance for all three metrics.

Apart from the Chan-Shelton market-making model in [6] that used reinforcement learning within the Glosten-Milgrom model, the Das market-making model in [8] is the closest model to ours under the same underlying market assumptions. More specifically, Das used Bayesian Learning to update a probability density function over the possible future prices. This probability density function forms the market-maker's belief about the future value of the fundamental price at every time-step. Although the Bayesian Learning is fully capable of tracking the fundamental value process over the long run, but it fails to immediately track this value following sudden and large jumps. This can potentially expose the market-maker to the risk of adverse selection. Our risk-sensitive RL framework in the trained region seem to cope better with such sudden changes in the fundamental value process. This can be seen in the performance in Figure 2 once the learning has converged (episodes 3250 and 3750 in particular).

In addition, we found that risk-sensitive reinforcement leaning algorithm driven by softmax action selection not only results in reduced variance for the three metrics, but also yields a better solution than the standard risk-neutral case. In particular, it creates a larger average profit than the standard case while maintaining a low inventory and a high spread. This is a good combination for a market-making strategy.

It is worth mentioning that the Softmax operator is prone to misbehaviour. This is primarily due to the fact that it is not a non-expansion operator for all parameter settings. Therefore convergence to a unique fixed point is not always guaranteed. However, in our numerical experiments, also based on figure (3), plot (d), the algorithm with the Softmax action selection rule converges. Although the choice of Softmax action selection rule is popular within the RL community, there are alternatives to the Softmax operator that are more stable and in some cases guarantee convergence. The reader is referred to [1] for a detailed treatment. In future work we will be analysing such operators with risk-sensitive economic agents.

The work reported here is not without limitations. In particular, because of the simplicity of the Glosten-Milgrom model on which the model is based, the market-making strategies are not learnt in an environment that includes a competitive order-driven market with multiple market-makers. In addition, the space of trading agents who are liquidity takers could be more realistic.

In future work, we will be analysing a more realistic simulation of the limit order book with multiple trading agents with mixture of strategies. [12] and [22], for example, proposed a more advanced information-based model of the market and [2], [3] and [11] adopted a similar information-based model of the market and analysed automated trading agents with different levels of information based on a continuous double auction system. We will adopt similar mechanisms to analyse automated market-making strategies driven by risk-sensitive reinforcement learning using a continuous double auction mechanism. Moreover [15] studied automated trading agents in a multi-agent-based setting, it is therefore worthwhile looking into multi-agent reinforcement leaning algorithms. This will help us investigate the interaction and competition between multiple market-makers using a double-sided auction. Finally, we will be implementing various risk-sensitive reinforcement learning with functional approximation techniques in order to study a larger state and action space.

REFERENCES

[1] K. Asadi and M. L. Littman. 2017. An alternative softmax operator for reinforcement learning. *In Proceedings of the 34th International Conference on Machine Learning* (2017), 243–252.

[2] D. Bloembergen, D. Hennes, P. McBurney, and K. Tuyls. 2015. Trading in markets with noisy information: An evolutionary analysis. *Connection Science.* (2015).

[3] D. Bloembergen, D. Hennes, S. Parsons, and K. Tuyls. 2015. Survival of the chartist: An evolutionary agent-based analysis of stock market trading. *In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems. AAMAS '15. Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.* (2015), 1699–1700.

[4] A. Brahma, M. Chakraborty, S. Das, A. Lavoie, and M. Magdon-Ismail. 2012. A Bayesian market maker. *In Proceedings of the 13th ACM Conference on Electronic Commerce* (2012), 215–232.

[5] J. S. Bridle. 1990. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimates of parameters. *In Advances in Neural Information Processing Systems* (1990).

[6] N. T. Chan. 2001. Artificial markets and intelligent agents. *PhD. Dissertation, Massachusetts Institute of Technology* (2001).

[7] N. T. Chan and C. R. Shelton. 2001. An electronic market-maker. *Technical Report AI Memo 2001-005, Massachusetts Institute of Technology, AI Lab* (2001).

[8] S. Das. 2005. A learning market-maker in the Glosten-Milgrom model. *Quantitative Finance* 5, 2 (2005), 169–180.

[9] S. Das and M. Magdon-Ismail. 2008. Adapting to market shock: Optimal sequential market-making. *In Proceedings of the 21st Annual Conference on Neural Information Processing Systems* (2008), 361–368.

[10] L. R. Glosten and P. R. Milgrom. 1985. Bid, ask and transaction prices in a specialist market with heterogeneously informed traders. *Journal of Financial Economics.* 14, 1 (1985), 71–100.

[11] D. Hennes, D. Bloembergen, M. Kaisers, K. Tuyls, and S. Parsons. 2012. Evolutionary advantage of foresight in markets. *In Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, GECCO '12. New York, NY, USA, 2012* (2012), 943–950.

[12] J. Huber, M. Kirchler, and M. Sutter. 2007. Is more information always better? Experimental financial markets with cumulative information. *Journal of Economic Behaviour and Organization.* (2007).

[13] Mihatsch O and R. Neuneier. 2002. Risk-sensitive reinforcement learning. *Machine Learning* 49 (2002), 267–290.

[14] M. O'Hara. 1995. *Market microstructure theory.* Blackwell, Cambridge.

[15] S. Phelps, K. Cai, P. McBurney, J. Niu, S. Parsons, and E. Sklar. 2008. Auctions, Evolution, and Multi-agent Learning. *Proceedings of the Symposium on Adaptive Learning Agents and Multi-Agent Systems* (2008), 188–210.

[16] G. Rummery and M. Niranjan. 1994. On-line Q-learning using connectionist systems. *Technical Report CUED/F-INFENG-TR 166, Cambridge University, UK.* (1994).

[17] C. R. Shelton. 2001. Importance sampling for reinforcement learning with multiple objectives. *PhD. Dissertation, Massachusetts Institute of Technology* (2001).

[18] S. P. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári. 2000. Convergence results for single-step on-policy reinforcement learning algorithms. *Machine Learning* (2000).

[19] S. P. Singh and R. S. Sutton. 1996. Reinforcement learning with replacing eligibility traces. *Machine Learning* (1996).

[20] R. S. Sutton. 1996. Generalisation in reinforcement learning: Successful examples using sparse coarse coding. *In Advances in Neural Information Processing Systems* (1996).

[21] R. S. Sutton and A. G. Barto. 2018. *Reinforcement Learning* (2nd. ed.). MIT Press, Cambridge, Massachusetts.

[22] B. Toth and E. Scalas. 2007. The value of information in financial markets: An agent-based simulation. *Information, Interaction, and (In)Efficiency in Financial Markets.* (2007).

[23] H. van Hasselt. 2010. Double Q-learning. *In Advances in Neural Information Processing Systems* (2010).

[24] H. van Hasselt. 2011. Insights in Reinforcement Learning: Formal Analysis and Empirical Evaluation of Temporal difference Learning. *SIKS dissertation series number 2011-04.* (2011).