Market Making via Reinforcement Learning

Thomas Spooner
Department of Computer Science
University of Liverpool
t.spooner@liverpool.ac.uk

John Fearnley
Department of Computer Science
University of Liverpool
john.fearnley@liverpool.ac.uk

Rahul Savani
Department of Computer Science
University of Liverpool
rahul.savani@liverpool.ac.uk

Andreas Koukorinis
Stratagem Technologies Ltd,
and University College London
andreas@stratagem.co

ABSTRACT

Market making is a fundamental trading problem in which an agent provides liquidity by continually offering to buy and sell a security. The problem is challenging due to inventory risk, the risk of accumulating an unfavourable position and ultimately losing money. In this paper, we develop a high-fidelity simulation of limit order book markets, and use it to design a market making agent using temporal-difference reinforcement learning. We use a linear combination of tile codings as a value function approximator, and design a custom reward function that controls inventory risk. We demonstrate the effectiveness of our approach by showing that our agent outperforms both simple benchmark strategies and a recent online learning approach from the literature.

KEYWORDS

Market Making; Limit Order Books; TD Learning; Tile Coding

ACM Reference Format:
Thomas Spooner, John Fearnley, Rahul Savani, and Andreas Koukorinis. 2018. Market Making via Reinforcement Learning. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 9 pages.

1 INTRODUCTION

The role of a market maker is to provide liquidity by facilitating transactions with other market participants. Like many trading problems, it has become increasingly automated since the advent of the electronic limit order book (LOB), as the need to handle more data and act on ever shorter time scales renders the task almost impossible for humans [25, 29]. Upwards of 60% of trading volume on some particularly active markets has been attributed to automated trading systems [27, 36]. This paper uses reinforcement learning (RL) to design competitive market making agents for financial markets using high-frequency historical equities data.

1.1 Related work.

Market making has been studied across a number of disciplines, including economics, finance, artificial intelligence (AI), and machine learning. A classic approach in the finance literature is to treat market making as a problem of *stochastic optimal control.* Here, a model for order arrivals and executions is developed and then control algorithms for the resulting dynamics are designed [3, 10, 11, 20, 23, 28]. Recent results in this line of research have studied price impact, adverse selection and predictability [1], and augmented the problem characteristics with risk measures and inventory constraints [7, 22].

Another prominent approach to studying market making and limit order book markets has been that of *zero-intelligence (ZI) agents.* The study of ZI agents has spanned economics, finance and AI. These agents do not “observe, remember, or learn”, but can, for example, adhere to inventory constraints [18]. Newer, more intelligent variants, now even incorporate learning mechanisms [14, 43]. Here, agents are typically evaluated in simulated markets without using real market data.

A significant body of literature, in particular in AI, has studied the market making problem for *prediction markets* [6, 34, 35]. In this setting, the agent’s main goal is to elicit information from informed participants in the market. While later studies have addressed profitability, the problem setup remains quite distinct from the financial one considered here.

Reinforcement learning has been applied for other financial trading problems [32, 37, 39], including optimal execution [33] and foreign exchange trading [16]. The first case of applying RL to market making [12] focused on the impact of noise (due to uninformed traders) on the agent’s quoting behaviour and showed that RL successfully converges on the expected strategies for a number of controlled environments. They did not, however, capture the challenges associated with explicitly handling order placement and cancellation, nor the complexities of using continuous state variables. Moreover, [12] found that temporal-difference RL struggled in their setting, a finding echoed in [39]. [12] attributed this to partial observability and excessive noise in the problem domain, despite the relative simplicity of their market simulation. In follow up work, [38] used importance sampling as a solution to the problems observed with off-policy learning. In contrast, we find temporal-difference RL to be effective for the market making problem, provided that we use eligibility traces and carefully design our function approximator and reward function.

One of the most recent related works is [2], which uses an *online learning* approach to develop a market making agent. They prove nice theoretical results for a stylized model, and empirically evaluate their agents under strong assumptions on executions. For example, they assume that the market has sufficient liquidity to execute market orders entirely at the posted price with no slippage. We use

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden. © 2018 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

this approach as one of the benchmarks for our empirical evaluation and address the impact of trading in a more realistic environment.

1.2 Our contributions.

The main contribution of this paper is to design and analyse temporal-difference (TD) reinforcement learning agents for market making. In contrast to past work [12, 38] we develop a high-fidelity simulation using high-frequency historical data. We identify eligibility traces as a solution to the unresolved issues previously associated with reward attribution, noise and partial observability [12]. We then design a reward function and state representation and demonstrate that these are key factors in the success of our final agent. We outline the steps taken to develop our agent below:

(1) We build a **realistic, data-driven simulation of a limit order book** using a basket of 10 equities across 5 venues and a mixture of sectors. This data includes 5 levels of order book depth and transaction updates with time increments on the order of milliseconds.
(2) We address concerns raised in past work about the efficacy of one-step temporal-difference learning, corroborating their results but demonstrating that **eligibility traces** are a simple and effective solution.
(3) We investigate the performance of a wide range of new and old TD-based learning algorithms.
(4) We show that the "natural" choice of reward function (incremental profit and loss) does not lead to the best performance and regularly induces instability during learning. We propose a solution in the form of an **asymmetrically dampened** reward function which improves learning stability, and produces higher and more consistent returns.
(5) We provide an evaluation of three different state space constructions and propose a **linear combination of tile codings** as our final representation as it gives competitive performance with more stable learning.
(6) We present a **consolidated agent**, based on a combination of the best results from 1–5 above, and show that it produces the best risk-adjusted out-of-sample performance compared to a set of simple benchmarks, a basic RL agent, and a recent online learning approach [2]. Moreover, we show that the performance of our consolidated agent is competitive enough to represent a potentially viable approach for use in practice.

2 PRELIMINARIES

2.1 Limit order books.

A *limit order* (LO) is an offer to buy or sell a given amount of an asset at a fixed price (or better). Each limit order specifies a *direction* (buy/sell or, equivalently, bid/ask), a *price* and a *volume* (how much to be traded). A *limit order book* is an aggregation of LOs that have been submitted to the market. The book has a fixed number of *price levels*, and the gap between the price levels is called the *tick size*. An example limit order book is shown in Fig. 1.

There are two types of orders: *Market Orders* (MOs) and *Limit Orders*. A market order is a request to buy or sell *now* at the current market price. When it arrives at the exchange it is executed against limit orders starting with the best price. That is, a sell market order will execute against the highest priced bid limit orders, and a buy market order will execute against the lowest priced ask limit orders. This process may spill-over to further price levels if there is not enough volume at the first level of the book.

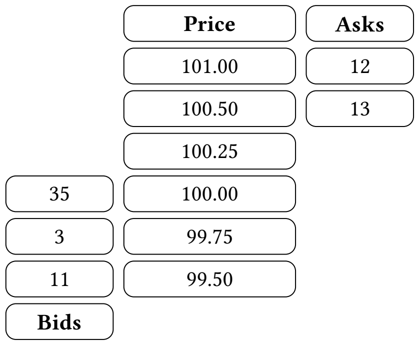

Figure 1: Snapshot of a limit order book with multiple price levels occupied by bid or ask orders and a total volume.

In contrast, a limit order is an offer to buy or sell at prices no worse than a limit price. When a limit order arrives at the market, it joins the back of the execution queue at its quoted price level, meaning that market orders will match against limit orders at a given price on a first-come first-served basis. If a limit order is submitted at a price level that currently holds limit orders with opposite direction (for example, a bid limit order submitted to a price level with ask limit orders quoted), then they are immediately executed against each other. This means that a given price level can only have one direction quoted at any given time.

The *market spread* is the difference between the best bid and ask quotes in the LOB. The *mid-price*, m, is the mid-point of the spread, defined simply by the arithmetic mean of the best bid and ask. When the best bid or ask changes, the *mid-price move*, denoted by Δm, is the change in the mid-price since the last time period. For example, in Fig. 1, the best bid and ask prices are 100.00 and 100.50, respectively, the mid-price is 100.25 and the market spread is 0.5. For a full examination of LOBs, their mechanics and nomenclature, see Gould et al. [2013] or Abergel et al. [2016].

2.2 Market making.

Market makers (MMs) are "traders who profit from facilitating exchange in a particular asset and exploit their skills in executing trades" [9]. They continually quote prices at which they are willing to trade on *both* sides of the book and, in so doing, provide liquidity to the market. If both orders execute, then they gain their *quoted spread*, which will be at least as big as the market spread, but may be more if they did not quote at the best price on either side.

Example. Consider a trader acting in the order book shown in Fig. 1, with ask and bid orders at 101.00 and 99.50, respectively, each for $q = 100$ units of volume. In this case, the quoted spread is $\Delta = 1.50$ and if both orders execute then the trader will make $q\Delta = 100 \times 1.50 = 150$ profit.

By quoting on both sides of the book, there is a risk that only one of the orders will execute, and thus the market maker may pick up a non-zero *inventory*. This may be positive or negative, and represents the extent to which the market maker has bought more than it has sold, or vice versa. A simple idealised market maker seeks to make money by repeatedly making its quoted spread, while keeping its inventory low to minimise market exposure. A more advanced market maker may choose to hold a non-zero inventory to exploit clear trends, whilst simultaneously capturing the spread.

3 THE SIMULATOR

We have developed a simulator of a financial market via direct reconstruction of the limit order book from historical data. This data comprised 10 securities from 4 different sectors, each with 8 months (January — August 2010) of data about the order-book depth, which records how much volume was at each level of the LOB, and the actual transactions that took place. The simulator tracks the top 5 price levels in the limit order book, and allows an agent to place its own orders within these price levels.

This approach to simulating an exchange is highly realistic and makes limited assumptions about the market. It should be noted that, since the market is reconstructed from historical data, simulated orders placed by an agent cannot impact the market. However, for all of the agents studied in this paper, the size of the orders used by the agent are small compared to the total amount traded in the market, so the impact of the agent's orders would be negligible.

One problem does arise, however, due the fact that we only have aggregate information about the limit orders. When we see that the amount of volume at a particular price level has decreased, we know that either some of the orders at that level have either been executed or they have been cancelled. Since we have transaction data, we can deal with the case where limit orders have been executed, but when limit orders are cancelled, we do not know precisely *which* orders have been removed.

This causes a problem when an agent's order is currently being simulated for that price level, because we do not know whether the cancelled order was ahead or behind the simulated order in the queue. Our solution is to assume that cancellations are distributed uniformly throughout the queue. This means that the probability that the cancelled order is ahead of the agent's order is proportional to the amount of volume ahead of the agent's order compared to the amount of volume behind it.

4 THE AGENT

4.1 Trading strategy.

In this paper we consider a state-based market making agent that acts on events as they occur in the LOB, subject to constraints such as upper and lower limits on inventory. An event may occur due to anything from a change in price, volume or arrangement of orders in the book; anything that constitutes an observable change in the state of the environment. Importantly, this means that the agent's actions are not spaced regularly in time. Since we are building a market maker, the agent is required to quote prices at which it is willing to buy and sell at all valid time points, unless the inventory constraints are no longer satisfied at which point trading is restricted to orders that bring the agent closer to a neutral position.

The action-space (Table 1) is based on a typical market making strategy in which the agent is restricted to a single buy and sell order and cannot exit the market, much like that considered by Chakraborty and Kearns [2011]. There are ten available actions, the first nine (IDs 0–8) of which correspond to a pair of orders with a particular spread and bias in their prices. Smaller values of $\theta_{\mathrm{a,b}}$ lead to quotes that are closer to the top of the book, while larger numbers cause the quotes to be deeper in the book. This means that the agent may choose to *quote wide or tight*, or even to *skew its orders* in favour of the buy/sell side. The final action (ID 9) allows the agent to *clear its inventory* using a market order.

Table 1: A typical action-space for market making.

Action ID	0	1	2	3	4	5	6	7	8
Ask (θ_{a})	1	2	3	4	5	1	3	2	5
Bid (θ_{b})	1	2	3	4	5	3	1	5	2
Action 9	MO with $\mathrm{Size}_{\mathrm{m}} = -\mathrm{Inv}(t_i)$								

Formally, actions 0 through 8 cause limit orders to be placed at fixed distances relative to a reference price, $\mathrm{Ref}(t_i)$, chosen here to be a measure of the true market price: the mid-price. At each time step, t_i, the agent revises two control parameters, $\theta_{\mathrm{a}}(t_i)$ and $\theta_{\mathrm{b}}(t_i)$, from which the relative prices, $\mathrm{Dist}_{\mathrm{a,b}}(t_i)$, are derived; the allowed values are given in Table 1. The full specification of the agent's pricing strategy is thus given by Eqs. 1 and 2 below,

$$p_{\mathrm{a,b}}(t_i) = \mathrm{Ref}(t_i) + \mathrm{Dist}_{\mathrm{a,b}}(t_i), \tag{1}$$

$$\mathrm{Dist}_{\mathrm{a,b}}(t_i) = \theta_{\mathrm{a,b}}(t_i) \cdot \mathrm{Spread}(t_i), \tag{2}$$

where $\mathrm{Spread}(t_i)$ is a time-dependent scale factor which must be a multiple of the market's tick size. In the experiments to follow, $\mathrm{Spread}(t_i)$ is calculated by taking a moving average of the market half-spread: $s(t_i)/2$.

All limit orders are placed with fixed, non-zero integer volumes, $\mathrm{Size}_{\mathrm{a,b}}$. The market order used in action 9 is sized proportionately to the agent's current inventory, $\mathrm{Size}_{\mathrm{m}}(t_i) = -\alpha \cdot \mathrm{Inv}(t_i)$, where $\alpha \in (0, 1]$ is a scaling factor and $\mathrm{Inv}(t_i)$ is the agent's inventory at time t_i. We ensure that $\mathrm{Size}_{\mathrm{m}}$ is restricted to be a non-zero integer multiple of the minimum lot size in the market. Note that volume is defined to be negative for buy orders and positive for sell orders, as in the nomenclature of Gould et al. [2013].

4.2 Reward functions.

The "natural" reward function for trading agents is profit and loss (PnL): i.e. how much money is made or lost through exchanges with the market. Under this reward function, the agent is encouraged to execute trades and hold an inventory which will appreciate in value over time; this is captured in Eq. 3.

Formally, for a given time t_i, let $\mathrm{Matched}_{\mathrm{a}}(t_i)$ and $\mathrm{Matched}_{\mathrm{b}}(t_i)$ be the amount of volume *matched* (*executed*) against the agent's orders since the last time t_{i-1} in the ask and bid books, respectively, and let $m(t_i)$ denote the mid-price at time t_i. We define

$$\psi_{\mathrm{a}}(t_i) \triangleq \mathrm{Matched}_{\mathrm{a}}(t_i) \cdot [p_{\mathrm{a}}(t_i) - m(t_i)]\,,$$

$$\psi_{\mathrm{b}}(t_i) \triangleq \mathrm{Matched}_{\mathrm{b}}(t_i) \cdot [m(t_i) - p_{\mathrm{b}}(t_i)]\,,$$

where $p_{\mathrm{a,b}}(t_i)$ are given by Eq. 1. These functions compute the money lost or gained through executions of the agent's orders *relative* to the mid-price. Observe that if both orders are executed in the same time interval $[t_{i-1}, t_i]$, then $\psi_a(t_i) + \psi_b(t_i)$ simplifies to the agent's quoted spread. Then, if $\text{Inv}(t_i)$ denotes the agent's inventory at time t_i, we can define the incremental (non-dampened) PnL function $\Psi(t_i)$ by setting $\Psi(t_0) \triangleq 0$ and

$$\Psi(t_i) \triangleq \psi_a(t_i) + \psi_b(t_i) + \text{Inv}(t_i)\Delta m(t_i). \tag{3}$$

The third (inventory) term here corresponds to the change in the agent's cash holdings due to exposure to changes in price in the market. Note that this term is only necessary because we accounted for the PnL from the agent's trades relative to the mid-price.

While the definition given above is a natural choice for the problem domain, it will be shown in this paper that such a basic formulation of the reward function ignores the specific objectives of a market maker, often leading to instability during learning and unsatisfactory out-of-sample performance. We thus study two alternative, *dampened* definitions of reward which are engineered to disincentivise trend-following and bolster spread capture. The three reward functions that we study are as follows:

PnL:

$$r_i = \Psi(t_i). \tag{4}$$

Symmetrically dampened PnL:

$$r_i = \Psi(t_i) - \eta \cdot \text{Inv}(t_i)\Delta m(t_i). \tag{5}$$

Asymmetrically dampened PnL:

$$r_i = \Psi(t_i) - \max[0, \eta \cdot \text{Inv}(t_i)\Delta m(t_i)]. \tag{6}$$

For the final two reward functions, the dampening is applied to the inventory term, using a scale factor η. Intuitively, this reduces the reward that the agent can gain from speculation. Specifically, the symmetric version dampens both profits and losses from speculation, while asymmetric dampening reduces the profit from speculative positions but keeps losses intact. In both cases, the amount of reward that can be gained from capturing the spread increases relative to the amount of reward that can be gained through speculation, thus encouraging market making behaviour.

4.3 State representation.

The state of the environment is constructed from a set of attributes that describe the condition of the agent and market, called the *agent-state* and *market-state*, respectively. The agent-state is described by three variables, the first is **inventory**, $\text{Inv}(t_i)$ — the amount of stock currently owned or owed by the agent; equivalently, one may think of the agent's inventory as a measure of *exposure*, since a large absolute position opens the agent to losses if the market moves. Next are the **active quoting distances**, normalised by the current spread scale factor, $\text{Spread}(t_i)$; these are the effective values of the control parameters, $\theta_{a,b}$, after stepping forward in the simulation.

The greatest source of state complexity comes from the market. Unlike the agent's internal features, the market state is subject to partial observability and may not have a Markovion representation. As such, the choice of variables, which may be drawn from the extensive literature on market microstructure, must balance expressivity with informational value to avoid Bellman's curse of dimensionality. In this paper, we include the following:

(1) Market (bid-ask) spread (s),
(2) Mid-price move (Δm),
(3) Book/queue imbalance,
(4) Signed volume,
(5) Volatility,
(6) Relative strength index.

Three different representations of state, based on these parameters, are considered. The first two, named the *agent-state* and *full-state*, are represented using a conventional tile coding approximation scheme. The third takes a more involved approach, whereby the agent-state, market-state and full-state are all approximated simultaneously using independent tile codings, and the value is a (fixed) linear combination of the values in these three tile codings. We refer to this as a linear combination of tile codings (LCTC).

Linear combination of tile codings. Assume there are N independent tile codings. Let $\hat{v}_i(x)$ be the value estimate of the state, x, given by the i-th tile coding. The value of a state is then defined by the following sum:

$$\hat{v}(x) = \sum_{i=0}^{N-1} \lambda_i \hat{v}_i(x) = \sum_{i=0}^{N-1} \lambda_i \sum_{j=0}^{n_i-1} b_{ij}(x)\, w_{ij}, \tag{7}$$

where n_i is the number of tiles in the i-th tile coding, w_{ij} is the j-th value of the associated weight vector and b_{ij} is the binary activation of the j-th tile (taking a value 0 or 1). We enforce that the λ_i weights (or influences) sum to unity for simplicity: $\sum_{i=0}^{N-1} \lambda_i = 1$.

This approach amounts to learning three independent value functions, each of which is updated using the same TD-error for gradient descent. Related approaches involve using multiple tile codings with varying granularity, but using the same set of state variables, and specific applications to dimension mapping for dynamics modelling [17, 21]. It has been argued that a coarse representation improves learning efficiency of a high resolution representation by directly the agent towards more optimal regions of policy space. The justification here is the same, the crucial difference is that we exploit the (approximate) *independency of variables* governing the agent-state and market-state rather than the shape of features; though these techniques could be combined.

This technique is particularly relevant for problems where a lot of domain-specific knowledge is available a priori. For example, consider a trading agent holding an inventory of, say, 100 units of a given stock. Though the expected future value of the holdings are surely conditional on the state of the market, the most important factor for the agent to consider is the risk associated with being exposed to unpredictable changes in price. Learning the value for the agent-, market- and full-state representations independently enables the agent to learn this much faster since it does not rely on observing every permutation of the full-state to evaluate the value of it's inventory. It is plausible that this will also help the agent converge (as in the granularity version mentioned previously) by guiding the agent away from local optima in policy space.

4.4 Learning algorithms.

We consider Q-learning (QL), SARSA and R-learning [41]. Several variants on these are also evaluated, including: Double Q-learning [26], Expected SARSA [24], On-policy R-learning and Double R-learning. Though other work has taken the approach of developing a specialist algorithm for the domain [33], these general purpose algorithms have been shown to be effective across

Table 2: Default parameters as used by the learning algorithm and the underlying trading strategy.

	Value
Training episodes	1000 days
Training sample size	$\sim$ 120 days
Testing sample size	40 days
Memory size	10^7
Number of tilings (M)	32
Weights for linear combination of tile codings [agent, market, full] (λ_i)	$(0.6, 0.1, 0.3)$
Learning rate (α)	0.001
Step-size [R-learning] (β)	0.005
Discount factor (γ)	0.97
Trace parameter (λ)	0.96
Exploration rate (ε)	0.7
$\varepsilon_{\text{Floor}}$	0.0001
ε_{T}	1000
Order size (ω)	1000
Min inventory (min Inv)	-10000
Max inventory (max Inv)	10000

a wide array of problems. Further, by including discounted and undiscounted learning methods we may identify characteristics of the problem domain that will guide our approach.

It should be assumed throughout this paper that each algorithm was implemented using eligibility traces (ETs), unless otherwise stated. As such, the conventional suffix, as used by Q(λ) versus Q-learning, shall be omitted in favour of uniformity[1].

5 RESULTS

We begin by introducing a set of benchmarks: a group of fixed "spread-based" strategies, and an online learning approach based on the work of Abernethy and Kale [2013] to anchor the performance of our agents in the literature. Next, a pair of *basic agents*, using an agent-state representation, non-dampened PnL reward function and QL or SARSA, will be presented. These basic agents represent the best attainable results using "standard techniques" and are closely related to those introduced by Chan and Shelton [2001]. We then move on to an assessment of our proposed *extensions*, each of which is evaluated and compared to the basic implementation. This gives an idea of how each of our modifications perform separately. Finally, we present a *consolidated agent* combining the most effective methods and show that this approach produces the best out-of-sample performance relative to the benchmarks, the basic agent, and in absolute terms.

Experimental setup. In each experiment, the dataset was split into disjoint training and testing sets, where all of the testing data occurs chronologically later than the training data; separate validation sets were used for hyperparameter optimisation and for drawing comparisons between the various agents considered in this paper. Unless otherwise stated, it should be assumed that each of these experiments were conducted using the parameters quoted in Table. 2 which were chosen through a combination of good practices and the use of a genetic algorithm (GA). The GA was specifically applied to find effective combinations of state variables, their lookback parameters and the weights used with the LCTC.

Performance criteria. The primary metric used to assess the performance of a trading strategy is the amount of money that it makes. While it is common to do this in terms of *returns* (final excess capital divided by starting capital), this does not make sense for a market maker since there is no natural notion of starting capital. We also cannot rate our strategies based on *profit* since each agent is tested across a variety of stocks, each of which have varying prices and liquidity. Instead, we introduce a *normalised daily PnL* that rates how good our strategies are at capturing the spread. This metric is defined on a daily basis as the total profit divided by the average market spread which normalises the profit across different markets; this amounts to the number of market spreads that would need to be captured to obtain that profit.

Ideally, market makers should not keep large inventories. To measure how well our agents achieve this, the *mean absolute position* (MAP) held by the agent is quoted as well. For example, high values for this metric may indicate that the agent has taken a speculative approach to trading. On the other hand, small values could suggest that the agent relies less on future changes in market value to derive it's earnings.

Uncertainties are quoted using the standard deviation and mean absolute deviation for normalised PnL and MAP, respectively.

5.1 Benchmarks.

To compare with existing work, we include a spread-based benchmark strategy which uses an online learning meta-algorithm; the original version, introduced by Abernethy and Kale [2013], was adapted to work in our simulator. We consider the *MMMW* variant which makes use of the multiplicative weights algorithm to pick in each period from a class of simple strategies parametrised by a minimum quoted spread[2]. The results, which are shown in Table 6, present less favourable performance than the original paper which found the strategy to be profitable over all dates and stocks considered. This could be attributed to the use of a less realistic market simulation that did not, for example, track the limit order book to the same level of precision considered here. This may indicate that their results do not generalise to more realistic markets.

A set of benchmarks were then developed using simple and random policies over the market making action space in Table 1. These strategies quote at fixed, symmetric distances from the chosen reference price ($\theta_{\text{a}} = \theta_{\text{b}}$) at all times. Unlike the previous benchmark strategy, this approach accounts for liquidity in the market by adapting it's prices to changes in the bid-ask spread. Further, in all of these strategies, market orders are used to forcibly clear the agent's inventory if the upper or lower bound is reached.

A sample of the results is given in Table 3 for `HSBA.L`; agent performance was found to be consistent across the full dataset. Fixed strategies with $\theta_{\text{a,b}} > 1$ were found to be just profitable on average, with decreasing MAP as $\theta_{\text{a,b}}$ increases. In all cases we find

[1]The code is provided open source at https://github.com/tspooner/rl_markets.

[2]Consistent with the original work, MMMW was found to outperform the follow-the-leader variant; see the extended version for a full performance summary [40].

Table 3: Out-of-sample normalised daily PnL (ND-PnL) and mean absolute position (MAP) for fixed and random benchmark strategies with inventory clearing, evaluated on HSBA.L.

	ND-PnL [10^4]	MAP [units]
Fixed ($\theta_{a,b}$ = 1)	−20.95 ± 17.52	3646 ± 2195
Fixed ($\theta_{a,b}$ = 2)	2.97 ± 13.12	3373 ± 2181
Fixed ($\theta_{a,b}$ = 3)	0.42 ± 9.62	2674 ± 1862
Fixed ($\theta_{a,b}$ = 4)	1.85 ± 10.80	2580 ± 1820
Fixed ($\theta_{a,b}$ = 5)	**2.80 ± 10.30**	2678 ± 1981
Random (Table 1)	−10.82 ± 5.63	**135 ± 234**

Table 4: Mean and standard deviation on the normalised daily PnL (given in units of 10^4) for Q-learning and SARSA using non-dampened PnL reward function and agent-state.

	QL	SARSA
CRDI.MI	**8.14 ± 21.75**	4.25 ± 42.76
GASI.MI	−4.06 ± 48.36	**9.05 ± 37.81**
GSK.L	4.00 ± 89.44	**13.45 ± 29.91**
HSBA.L	**−12.65 ± 124.26**	−12.45 ± 155.31
ING.AS	−67.40 ± 261.91	**−11.01 ± 343.28**
LGEN.L	**5.13 ± 36.38**	2.53 ± 37.24
LSE.L	4.40 ± 16.39	**5.94 ± 18.55**
NOK1V.HE	**−7.65 ± 34.70**	−10.08 ± 52.10
SAN.MC	−4.98 ± 144.47	**39.59 ± 255.68**
VOD.L	**15.70 ± 43.55**	6.65 ± 37.26

that the strategy suffers from high volatility. This may have been caused by a lack of proper inventory management, as indicated by the consistently high mean average positions.

5.2 Basic agent.

The basic agent is used to bound the expected performance *without* the extensions proposed in this paper. This agent used a state representation comprising only of the agent-state — inventory and normalised bid/ask quoting distances — and the non-dampened PnL reward function (Eq. 4). This agent was trained using one-step Q-learning and SARSA and, consistent with the findings of past work [12], were unable to obtain useful results in either case. The addition of eligibility traces (known as Q(λ) and SARSA(λ)) improved the agents' performance, yielding strategies that occasionally generated profits out-of-sample as seen in Table 4. These results, however, were still unsatisfactory.

5.3 Extensions.

In this section, three extensions to the basic agent are discussed: variations on the learning algorithm, reward function and state representation. The results are given in Table 5. Each row of the table consists of the basic agent, *again only agent variables included in the state space*, with the one modifier. SARSA with eligibility traces is used as base the learning algorithm, unless stated otherwise.

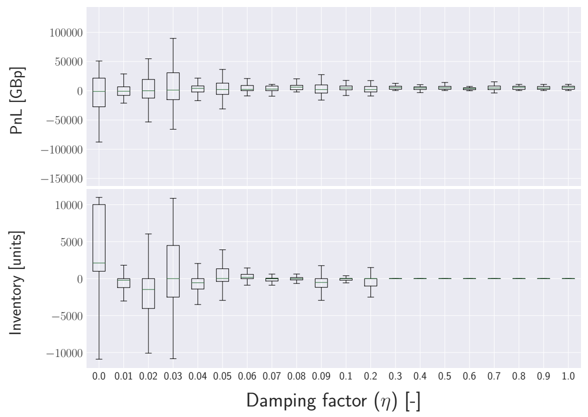

Figure 2: Distributions of daily out-of-sample PnL and mean inventory for increasing values of the damping factor, η, evaluated on HSBA.L.

Learning algorithms. The first variation on the basic agent deals with the impact of using different learning algorithms to QL and SARSA; the findings are summarised in Table 5. It was found that variants based on off-policy learning tended to perform worse than their on-policy counterparts. Note, for example, the difference between the consistency and outright performance between R-learning and on-policy R-learning. This may be attributed to known divergence issues with classical off-policy methods.

Though variants were found to outperform the basic agent for some specific equity, none were as consistent as SARSA. In the majority of cases they performed worse, both in out-of-sample performance and learning efficiency. For example, Double Q-learning, despite mitigating the maximisation bias [26] versus regular Q-learning, was found to be less profitable and more volatile on most assets; save for CRDI.MI, a highly active and volatile stock which may benefit from the double-estimator approach. This again suggests that, while each stock could be optimised individually for maximal performance, SARSA may be used reliably as a baseline.

Reward functions. The results in Table 5 show that the intuitive choice of reward function (non-dampened PnL) does not equate to the best out-of-sample performance across the basket of securities. Though symmetric dampening was found to exacerbate the flaws in the basic agent, asymmetric dampening of the trend-following term in Eq. 3, with sufficiently high damping factor (η), was found to produce superior risk-adjusted performance in most cases. This is exemplified by Fig. 2 which shows the distribution of out-of-sample (actual) PnL for increasing values of the damping factor.

Fig. 2 suggests that there is a value of $\eta \sim 0.1$ about which the agent begins to converge on fundamentally different policies than those promoted by the non-dampened PnL function (Eq. 3). This shift in the final policy is manifested in the change from holding large, biased inventories towards small and neutral positions. This reduction in exposure, beginning from $\eta \sim 0.05$, comes along with a change in PnL, which also tends towards lower variance as a result. Though the precise value of η varies across the basket of securities, the impact of asymmetric dampening is clear, providing strong evidence that the inventory term in Eq. 3 is the leading driver of agent

Table 5: Mean and standard deviation on the normalised daily PnL (given in units of 10^4) for various extensions to the basic agent, each evaluated over the basket of stocks.

	CRDI.MI	GASI.MI	GSK.L	HSBA.L	ING.AS	LGEN.L	LSE.L	NOK1V.HE	SAN.MC	VOD.L
Double Q-learning	−5.04 ± 83.90	5.46 ± 59.03	6.22 ± 59.17	5.59 ± 159.38	58.75 ± 394.15	2.26 ± 66.53	16.49 ± 43.10	−2.68 ± 19.35	5.65 ± 259.06	7.50 ± 42.50
Expected SARSA	0.09 ± 0.58	3.79 ± 35.64	−9.96 ± 102.85	25.20 ± 209.33	6.07 ± 432.89	2.92 ± 37.01	6.79 ± 27.46	−3.26 ± 25.60	32.28 ± 272.88	15.18 ± 84.86
R-learning	5.48 ± 25.73	−3.57 ± 54.79	12.45 ± 33.95	−22.97 ± 211.88	−244.20 ± 306.05	−3.59 ± 137.44	8.31 ± 23.50	−0.51 ± 3.22	8.31 ± 273.47	32.94 ± 109.84
Double R-learning	19.79 ± 85.46	−1.17 ± 29.49	21.07 ± 112.17	−14.80 ± 108.74	5.33 ± 209.34	−1.40 ± 55.59	6.06 ± 25.19	2.70 ± 15.40	32.21 ± 238.29	25.28 ± 92.46
On-policy R-learning	0.00 ± 0.00	4.59 ± 17.27	14.18 ± 32.30	9.56 ± 30.40	18.91 ± 84.43	−1.14 ± 40.68	5.46 ± 12.54	0.18 ± 5.52	25.14 ± 143.25	16.30 ± 32.69
Symm. Damp. ($\eta = 0.6$)	12.41 ± 143.46	9.07 ± 68.39	30.04 ± 135.89	−11.80 ± 214.15	90.05 ± 446.09	5.54 ± 119.86	8.62 ± 27.23	−4.40 ± 84.93	27.38 ± 155.93	8.87 ± 93.14
Asymm. Damp. ($\eta = 0.6$)	0.08 ± 2.21	−0.10 ± 1.04	9.59 ± 10.72	13.88 ± 10.60	−6.74 ± 68.80	4.08 ± 7.73	1.23 ± 1.80	0.52 ± 3.29	5.79 ± 13.24	9.63 ± 6.94
Full-state	−31.29 ± 27.97	−35.83 ± 13.96	−31.29 ± 27.97	−84.78 ± 31.71	−189.81 ± 68.31	−14.39 ± 9.38	−6.76 ± 11.52	−9.30 ± 23.17	−144.70 ± 104.64	−21.76 ± 17.71
LCTC-state	−5.32 ± 52.34	5.92 ± 40.65	5.45 ± 40.79	−0.79 ± 68.59	9.00 ± 159.91	6.73 ± 22.88	3.04 ± 5.83	−2.72 ± 19.23	52.55 ± 81.70	7.02 ± 48.80

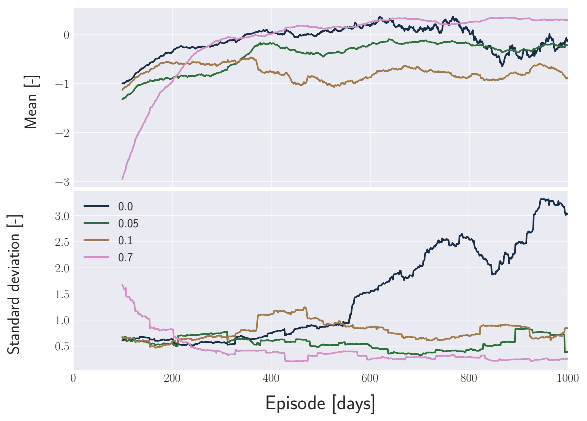

Figure 3: Rolling mean and standard deviation on the average episodic reward during training for increasing values of the damping factor, η, evaluated on HSBA.L.

behaviour and should be manipulated to match one's risk profile. This result relates strongly with the use of inventory penalising terms in the value function in stochastic optimal control [7, 8] and is a key contribution of this paper.

It was also observed that asymmetric dampening of the reward function lead to improved stability during learning and better asymptotic performance compared to the basic agent. Fig. 3 shows how the mean and standard deviation of episodic reward varies with increasing values of the damping factor, η, during training. The change due to dampening is particularly apparent from the standard deviation which is seen to diverge for $\eta = 0.0$ and decrease for $\eta \geq 0.1$. This suggests that the inventory component in Eq. 3 not only drives behaviour, but is also the main source of instability, where increasing values of η appear to yield better and more consistent performance. It is plausible that the large scale and variation in rewards due to changes in the agent's position dominates the spread-capture term, and makes it very difficult to learn an accurate estimate of the value function; one solution could be to model the value function itself using a distribution [5].

State representation. Three state representations were considered: an *agent-state* (as used by the basic agent), a *full-state* (agent and market variables) and a *linear combination of tile codings.* The objective was to identify and resolve challenges associated with increased

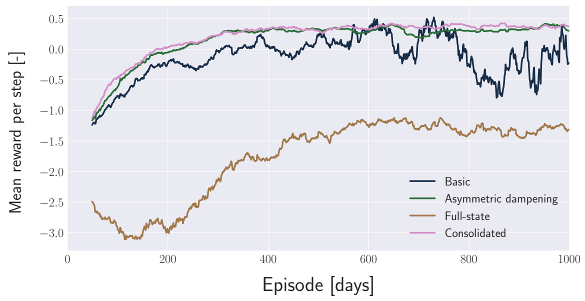

Figure 4: Rolling mean of the average episodic reward for different agent variants during training on HSBA.L.

state complexity when including market parameters, which, as seen by the performance of the full-state variant (Table 5), were found to have a strongly negative impact on returns. Contrary to intuition, we did not observe any considerable improvement in performance with increased training, instead, the agent was regularly seen to degrade and even diverge (both in episodic reward and TD-error).

Fig. 4 shows that the basic agent is more efficient at learning than the full-state variant and tends to be more stable; this is to be expected since the state-space is much smaller. To capitalise on this stability and efficiency, whilst incorporating market information, we turn to a *linear combination of tile codings (LCTC).* As shown in Table 5, this variant considerably outperforms the full-state agent and doesn't appear to suffer from issues of divergence as seen in with the full-state variant. Though in some cases it produced lower PnL than the basic agent (as for GSK.L), this does not indicate a problem with the approach, rather it is related to the choice of the variables used to describe the state of the market; for example, we find that it performs particularly well on SAN.MC. Indeed the choice of which parameters to use for a given stock is a non-trivial task in an of itself. The key result is that the LCTC marries expressivity and efficiency and helps to prevent divergence even when the market variables have little informational content.

5.4 Consolidated agent.

Finally, a consolidation of the best variants on the basic agent is considered. It uses the asymmetrically dampened reward function with a LCTC state-space, trained using SARSA. In the majority of cases it was found that this agent generated slightly lower returns

Table 6: Comparison of the out-of-sample normalised daily PnL (ND-PnL) and mean absolute positions (MAP) of the benchmark strategies against the final presented reinforcement learning agent.

	Abernethy and Kale (MMMW) Benchmark		Fixed ($\theta_{a,b} = 5$) Benchmark		Consolidated Agent	
	ND-PnL [10^4]	MAP [units]	ND-PnL [10^4]	MAP [units]	ND-PnL [10^4]	MAP [units]
CRDI.MI	−1.44 ± 22.78	7814 ± 1012	−0.14 ± 1.63	205 ± 351	**0.15 ± 0.59**	1 ± 2
GASI.MI	−1.86 ± 9.22	5743 ± 1333	**0.01 ± 1.36**	352 ± 523	0.00 ± 1.01	33 ± 65
GSK.L	−3.36 ± 13.75	8181 ± 1041	0.95 ± 2.86	1342 ± 1210	**7.32 ± 7.23**	57 ± 105
HSBA.L	1.66 ± 22.48	7330 ± 1059	2.80 ± 10.30	2678 ± 1981	**15.43 ± 13.01**	104 ± 179
ING.AS	−6.53 ± 41.85	7997 ± 1265	**3.44 ± 23.24**	2508 ± 1915	−3.21 ± 29.05	10 ± 20
LGEN.L	−0.03 ± 11.42	5386 ± 1297	0.84 ± 2.45	986 ± 949	**4.52 ± 8.29**	229 ± 361
LSE.L	−2.54 ± 4.50	4684 ± 1507	0.20 ± 0.63	382 ± 553	**1.83 ± 3.32**	72 ± 139
NOK1V.HE	−0.97 ± 8.20	5991 ± 1304	**−0.52 ± 4.16**	274 ± 497	−5.28 ± 33.42	31 ± 62
SAN.MC	−2.53 ± 26.51	8865 ± 671	1.52 ± 11.64	3021 ± 2194	**5.67 ± 13.41**	4 ± 9
VOD.L	1.80 ± 22.83	7283 ± 1579	1.26 ± 4.60	1906 ± 1553	**5.02 ± 6.35**	46 ± 87

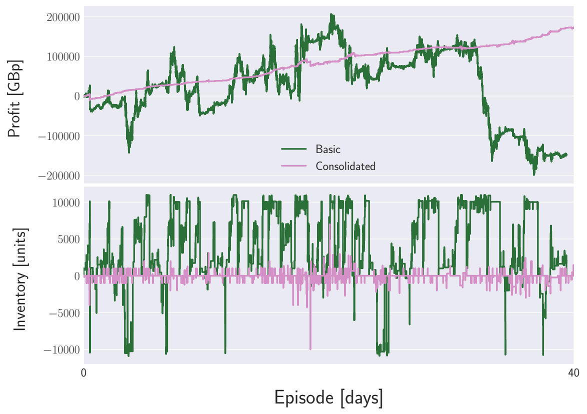

Figure 5: Out-of-sample equity curve and inventory process for the basic and consolidated agents, evaluated on HSBA.L.

than the best individual variants, but had improved out-of-sample stability; see Table 6. In addition, they tended to hold smaller inventories, which may have been a contributing factor towards the reduced uncertainty on PnL. Though the results vary slightly across the basket of securities, the consolidated agent was found to produce superior risk-adjusted performance over the basic agent and extended variants overall.

For example, Fig. 5 compares the equity and inventory processes for the basic and consolidated agents' out-of-sample tests. Both illustrate a profound difference in behaviour between the two. Where the former is highly volatile, the latter is stable. The basic agent regularly holds a non-zero inventory, exposing itself to changes in the security's value for extended periods of time, leading to the noise observed in the equity curve. For the consolidated agent, it appears that the learnt policy targets a near-zero inventory, relying less on speculative trading and thus yielding the consistency one expects from a good market making strategy.

6 CONCLUSIONS

We have developed a reinforcement learning agent that produces competitive out-of-sample performance across a basket of securities. We first developed a highly realistic simulation of the problem domain and showed how eligibility traces solve the problems raised in past works. We then investigated different learning algorithms, reward functions and state representations, and consolidated the best techniques into a single agent which is shown to produce superior risk-adjusted performance. The following represent key areas for future research based on our results:

(1) Apply more advanced learning algorithms such as GreedyGQ, Q(σ) and true online variants [15, 30, 42], which provide guarantees of convergence with linear function approximation, options learning [4] and direct policy search methods.
(2) Explore deep reinforcement learning, in particular using recurrent neural networks that should be well-suited to the sequential nature of the problem.
(3) Introduce parametrised action spaces [31] as an alternative to discrete action sets.
(4) Extend to multiple order and variable order size action spaces.
(5) Investigate the impact of market frictions such as rebates, fees and latency on the agent's strategy.
(6) Use sequential Bayesian methods [13] for better order book reconstruction and estimation of order queues.

REFERENCES

[1] Frederic Abergel, Anirban Chakraborti, Aymen Jedidi, Ioane Muni Toke, and Marouane Anane. 2016. *Limit Order Books*. Cambridge University Press.
[2] Jacob Abernethy and Satyen Kale. 2013. Adaptive Market Making via Online Learning. In *Proc. of NIPS*. 2058–2066.
[3] Marco Avellaneda and Sasha Stoikov. 2008. High-frequency trading in a limit order book. *Quantitative Finance* 8, 3 (2008), 217–224.
[4] Pierre-Luc Bacon, Jean Harb, and Doina Precup. 2016. The Option-Critic Architecture. *CoRR* abs/1609.05140 (2016).
[5] Marc G Bellemare, Will Dabney, and Rémi Munos. 2017. A Distributional Perspective on Reinforcement Learning. In *International Conference on Machine Learning*. 449–458.
[6] Aseem Brahma, Mithun Chakraborty, Sanmay Das, Allen Lavoie, and Malik Magdon-Ismail. 2012. A Bayesian market maker. In *Proc. of EC*. New York, New York, USA, 215–232.
[7] Álvaro Cartea and Sebastian Jaimungal. 2015. Risk Metrics and Fine Tuning of High-Frequency Trading Strategies. *Mathematical Finance* 25, 3 (2015), 576–611.
[8] Álvaro Cartea, Sebastian Jaimungal, and Damir Kinzebulatov. 2016. Algorithmic Trading with Learning. *International Journal of Theoretical and Applied Finance* 19, 04 (2016), 1650028.
[9] Álvaro Cartea, Sebastian Jaimungal, and José Penalva. 2015. *Algorithmic and High-Frequency Trading*. Cambridge University Press.
[10] Álvaro Cartea, Sebastian Jaimungal, and Jason Ricci. 2014. Buy Low Sell High: A High Frequency Trading Perspective. *SIAM Journal on Financial Mathematics* 5, 1 (2014), 415–444.
[11] Tanmoy Chakraborty and Michael Kearns. 2011. Market Making and Mean Reversion. In *Proc. of EC*. 307–314.
[12] Nicholas T. Chan and Christian R. Shelton. 2001. *An Electronic Market-Maker*. AI Memo 2001-005. MIT AI Lab.
[13] Hugh L Christensen, Richard E Turner, Simon I Hill, and Simon J Godsill. 2013. Rebuilding the limit order book: sequential Bayesian inference on hidden states. *Quantitative Finance* 13, 11 (2013), 1779–1799.
[14] Dave Cliff. 2006. ZIP60: an enhanced variant of the ZIP trading algorithm. In *CEC/EEE*.
[15] Kristopher De Asis, J Fernando Hernandez-Garcia, G Zacharias Holland, and Richard S Sutton. 2017. Multi-step reinforcement learning: A unifying algorithm. *arXiv preprint arXiv:1703.01327* (2017).
[16] M. A H Dempster and V. Leemans. 2006. An automated FX trading system using adaptive reinforcement learning. *Expert Systems with Applications* 30, 3 (2006), 543–552.
[17] Pei-yong Duan and Hui-he Shao. 1999. Multiple Hyperball CMAC Structure for Large Dimension Mapping. *Proc. of IFAC* 32, 2 (1999), 5237–5242.
[18] Dhananjay K. Gode and Shyam Sunder. 1993. Allocative Efficiency of Markets with Zero-Intelligence Traders: Market as a Partial Substitute for Individual Rationality. *Journal of Political Economy* 101, 1 (1993), 119–137.
[19] Martin D Gould, Mason A Porter, Stacy Williams, Mark McDonald, Daniel J Fenn, and Sam D. Howison. 2013. Limit order books. *Quantitative Finance* 13, 11 (2013), 1709–1742.
[20] Sanford J Grossman and Merton H Miller. 1988. Liquidity and Market Structure. *The Journal of Finance* 43, 3 (1988), 617.
[21] Marek Grzes and Daniel Kudenko. 2010. Reward Shaping and Mixed Resolution Function Approximation. In *Developments in Intelligent Agent Technologies and Multi-Agent Systems*. IGI Global, Chapter 7.
[22] Olivier Guéant, Charles Albert Lehalle, and Joaquin Fernandez-Tapia. 2013. Dealing with the inventory risk: A solution to the market making problem. *Mathematics and Financial Economics* 7, 4 (2013), 477–507.
[23] Fabien Guilbaud and Huyen Pham. 2011. Optimal High Frequency Trading with limit and market orders. *CoRR* abs/1106.5040 (2011).
[24] Seijen Harm van, Hasselt Hado van, Shimon Whiteson, and Marco Wiering. 2009. A theoretical and empirical analysis of expected SARSA. In *Proc. of ADPRL*. 177–184.
[25] Joel Hasbrouck and Gideon Saar. 2013. Low-latency trading. *Journal of Financial Markets* 16, 4 (2013), 646–679.
[26] Hado V Hasselt. 2010. Double Q-learning. In *Proc. of NIPS*. 2613–2621.
[27] Richard Haynes and John S Roberts. 2015. Automated Trading in Futures Markets. *CFTC White Paper* (2015).
[28] Thomas Ho and Hans R. Stoll. 1981. Optimal dealer pricing under transactions and return uncertainty. *Journal of Financial Economics* 9, 1 (1981), 47–73.
[29] M. Leaver and T. W. Reader. 2016. Human Factors in Financial Trading: An Analysis of Trading Incidents. *Human Factors* 58, 6 (2016), 814–832.
[30] Hamid Reza Maei, Csaba Szepesvari, Shalabh Bhatnagar, and Richard Sutton. 2010. Toward Off-Policy Learning Control with Function Approximation. *Proc. of ICML*, 719–726.
[31] Warwick Masson and George Konidaris. 2016. Reinforcement Learning with Parameterized Actions. *Proc. of AAAI*, 1934–1940.
[32] John E Moody and Matthew Saffell. 1998. Reinforcement Learning for Trading. In *Proc. of NIPS*. 917–923.
[33] Yuriy Nevmyvaka, Yi Feng, and Michael Kearns. 2006. Reinforcement learning for optimized trade execution. In *Proc. of ICML*. 673–680.
[34] Abraham Othman. 2012. *Automated Market Making: Theory and Practice*. Ph.D. Dissertation. Carnegie Mellon University.
[35] Abraham Othman, David M Pennock, Daniel M Reeves, and Tuomas Sandholm. 2013. A Practical Liquidity-Sensitive Automated Market Maker. *ACM Transactions on Economics and Computation* 1, 3 (2013), 1–25.
[36] Rahul Savani. 2012. High-frequency trading: The faster, the better? *IEEE Intelligent Systems* 27, 4 (2012), 70–73.
[37] L. Julian Schvartzman and Michael P. Wellman. 2009. Stronger CDA Strategies through Empirical Game-Theoretic Analysis and Reinforcement Learning. *Proc. of AAMAS* (2009), 249–256.
[38] C R Shelton. 2001. *Importance Sampling for Reinforcement Learning with Multiple Objectives*. Ph.D. Dissertation. Massachusetts Institute of Technology.
[39] Alexander A. Sherstov and Peter Stone. 2004. Three Automated Stock-Trading Agents: A Comparative Study. In *Agent Mediated Electronic Commerce {VI}: Theories for and Engineering of Distributed Mechanisms and Systems (AMEC 2004)*. Vol. 3435. 173–187.
[40] Tom Spooner, John Fearnley, Rahul Savani, and Andreas Koukorinis. 2018. Market Making via Reinforcement Learning. (2018). arXiv:1804.04216
[41] R.S. Sutton and A.G. Barto. 1998. Reinforcement Learning: An Introduction. *IEEE Transactions on Neural Networks* 9, 5 (1998), 1054–1054.
[42] Harm Van Seijen, A Rupam Mahmood, Patrick M Pilarski, Marlos C Machado, and Richard S Sutton. 2016. True online temporal-difference learning. *Journal of Machine Learning Research* 17, 145 (2016), 1–40.
[43] Perukrishnen Vytelingum, Dave Cliff, and Nicholas R Jennings. 2008. Strategic bidding in continuous double auctions. *Artif. Intell.* 172, 14 (2008), 1700–1729.