Law & Economics Working Papers

1-1-2022

Learning to Manipulate a Financial Benchmark

Megan Shearer
University of Michigan - Ann Arbor
Gabriel V. Rauterberg
University of Michigan Law School, rauterb@umich.edu
Michael P. Wellman
University of Michigan - Ann Arbor

Working Paper Citation

Shearer, Megan; Rauterberg, Gabriel V.; and Wellman, Michael P., "Learning to Manipulate a Financial Benchmark" (2022). *Law & Economics Working Papers*. 235.
https://repository.law.umich.edu/law_econ_current/235

Learning to Manipulate a Financial Benchmark

Megan Shearer[1], Gabriel Rauterberg[2], and Michael P. Wellman[1]

[1]Computer Science & Engineering, University of Michigan, Ann Arbor
[2]Law, University of Michigan, Ann Arbor
{*shearerj,rauterb,wellman*}*@umich.edu*

September 14, 2022

Abstract

Financial benchmarks estimate market values or reference rates used in a wide variety of contexts, but are often calculated from data generated by parties who have incentives to manipulate these benchmarks. Since the London Interbank Offered Rate (LIBOR) scandal in 2011, market participants, scholars, and regulators have scrutinized financial benchmarks and the ability of traders to manipulate them. We study the impact on market welfare of manipulating transaction-based benchmarks in a simulated market environment. Our market consists of a single benchmark manipulator with external holdings dependent on the benchmark, and numerous background traders unaffected by the benchmark. We explore two types of manipulative trading strategies: zero-intelligence strategies and strategies generated by deep reinforcement learning. Background traders use zero-intelligence trading strategies. We find that the total surplus of all market participants who are trading increases with manipulation. However, the aggregated market surplus decreases for all trading agents, and the market surplus of the manipulator decreases, so the manipulator's surplus from the benchmark significantly increases. This entails under natural assumptions that the market and any third parties invested in the opposite side of the benchmark from the manipulator are negatively impacted by this manipulation.

JEL Classification: C63; G17; G18.
Keywords: financial benchmark manipulation; algorithmic trading; agent-based modeling; deep reinforcement learning

1. Introduction

A financial market benchmark is a summary statistic over market variables, such as prices of specified securities at designated times. Benchmarks are employed by market participants for various purposes, including as reference measures for asset values (e.g., the S&P 500), interest rates (LIBOR), and market volatility (VIX); to define derivative instruments; or as price terms in contracts (Gellasch and Nagy, 2019). Benchmarks in the form of reference measures can provide a concise reflection of market realities, thereby supporting decision making in the real economy. As such, accurate benchmark prices constitute a positive externality from functional financial markets (Bond et al., 2012). Their use in financial instruments and contracts also serves a valuable function in commerce and risk management.

Given their role in market decisions and contracts, some entities may have stakes in benchmark values, and hence incentives to try to influence or *manipulate* them. For instance, the *London Interbank Offered Rate* (LIBOR), an estimate of the rate at which banks can borrow from each other, supports more than $300 trillion worth of loans around the world (McBride, 2016). Several major banks have been implicated in schemes to manipulate LIBOR in the last decade, and criminal charges have been brought against over twenty individuals in the U.S. and U.K. since 2015 (McBride, 2016). February 2018 saw accusations of manipulation in the Chicago Board Options Exchange (CBOE) *Volatility Index* (VIX), a measure of U.S. stock market volatility based on the cost of buying certain options (Banerji, 2018). LIBOR has been particularly vulnerable to manipulation because it is calculated using self-reported data provided by parties with conflicts of interest regarding the benchmark's value (Duffie and Dworczak, 2018; Gellasch and Nagy, 2019). In the wake of the LIBOR scandal, regulators, academics, and market participants lobbied for a transaction-based replacement for LIBOR, such as the Secured Overnight Finance Rate (SOFR) or the U.S. Dollar Intercontinental Exchange (ICE) Bank Yield Index (Duffie and Dworczak, 2018; ICE Benchmark Administration Limited, 2019). Whereas it may be harder to manipulate transaction-based benchmarks, it is still possible, as in the alleged manipulation of the VIX in 2018 and the World Markets/Reuters Closing Spot Rates (WM/R FX rates) in 2014 (Boyle, 2014).

Prior work has employed theoretical models and historical data to study benchmark manipulation in financial markets (Bariviera et al., 2016; Duffie and Dworczak, 2018; Duffie, 2018; Eisl et al., 2017; Rauch et al., 2013). Using a simulated market allows us to incorporate complex details of *market microstructure*, representing the actual mechanics of trade, interactions among market participants, and the structure of the market. By combining the agent-based model with game-theoretic reasoning, we can also consider the response of strategic agents to the presence of a benchmark manipulator, and consider a wide range of market settings, benchmark designs, and trading strategy options.

Our model employs a standard market mechanism organized around a limit order book for a single security. We assume a benchmark defined by transaction prices in this market. Trading agents may submit buy and sell orders, with orders executing immediately when matched, otherwise resting in the order book pending execution against a subsequent order. The market includes a manipulator agent, with external holdings of a contract tied to the benchmark. The rest of the market comprises background agents who have private reasons to trade, and a market maker who seeks profit by connecting these traders across time.

We consider three types of manipulation strategies. The first is a simple hand-crafted strategy, which extends the behavior of simple background traders by adjusting offers systematically in order to influence the benchmark in a certain direction. The other two types of benchmark manipulator generate their trading strategies through *deep reinforcement learning* (DRL). The two types correspond to qualitatively different RL algorithms, called *deep Q-network* (DQN) (Mnih et al., 2015) and *deep deterministic policy gradient* (DDPG) (Lillicrap et al., 2016). In both cases, the agent is not explicitly instructed to manipulate, but rather learns a policy (mapping from market state to orders submitted) that effectively achieves manipulation. These policies are derived through simulated experience with the market model, given a reward function that credits the agent for its profits from the market combined with profits from its contract holdings tied to the benchmark.

We determine the impact of benchmark manipulation by comparing market outcomes with and without manipulation. These comparisons reflect strategic responses of the background traders to the presence or absence of the manipulator. We find across a variety of settings that manipulation is profitable overall to the manipulators. The manipulation activity itself is costly, in that the manipulator must sacrifice trading profit to move the benchmark. The background traders actually benefit from the manipulation, as their aggregate gains from trading increase. The external parties dependent on the opposite side of the benchmark are the real losers from the manipulation, with their losses captured in part by the manipulator and in part by the background agents whose trading is effectively subsidized.

The key contributions of this paper are:

- A model of financial benchmark manipulation, instantiated in an agent-based simulation environment.
- Trading strategies that effectively and profitably manipulate the benchmark in this model, including techniques for automatically generating manipulation strategies using deep reinforcement learning. We demonstrate successful learning to manipulate using two qualitatively different RL algorithms, with and without the presence of market makers.
- Analysis of the impact of benchmark manipulation on market efficiency and on the welfare of respective participants, accounting for some variation in market structure and strategic response.

This paper is organized as follows. Following a discussion of related work in the next section, we describe the market environment in Section 3. Section 4 introduces the benchmark manipulator and methods for learning to manipulate. Section 5 describes our experimental design, and Section 6 presents the results and our analysis of the effect of benchmark manipulation. As the ability to automatically generate manipulation strategies presents significant new challenges for financial regulation, Section 7 provides commentary on how this study can inform policy. We conclude in Section 8.

2. Related Work

Martínez-Miranda et al. (2016) studied market manipulation using a Markov decision model, identifying conditions that are relatively favorable for manipulative strategies. Wang et al. (2021) developed an agent-based model of market manipulation, demonstrating settings where a spoofer can effectively influence market prices despite the presence of rationally responding traders. The current work builds on this agent-based approach, employing a similar market model extended to include a financial benchmark.

Mizuta (2020) showed that a genetic algorithm combined with agent-based simulation can learn a sequence of actions that profits in a specified simulation scenario by influencing the prices offered by other trading agents following a fixed market-sensitive strategy.

Significant attention has been paid to the potential of automating market manipulation through misinformation campaigns, in social media and other forums. Yagemann et al. (2021) study the potential for conducting *market-based* manipulation at scale, through botnet hijacking of brokerage accounts. On the basis of SEC data and agent-based simulation, they find that such attacks appear to be quite feasible.

The majority of prior research on *benchmark manipulation* is either theoretical or based on analysis of historical market data. Duffie and Dworczak (2018) introduce a theoretical model to analyze the robustness and bias of alternative benchmark constructions, and find that *volume-weighted average price* (VWAP) is optimal among linear benchmarks. Duffie (2018) also considers robustness to manipulation in design of an auction mechanism to convert LIBOR-based contracts to employ the replacement SOFR benchmark.

Bariviera et al. (2016) and Eisl et al. (2017) use historical data to find instances of manipulation of interest-rate benchmarks and provide suggestions for more robust benchmarks and regulation. Rauch et al. (2013) also use historical data to find instances of benchmark manipulation in LIBOR and investigate which banks were potentially involved in the 2011 scandal. Griffin and Shams (2018) examine spikes at time of settlement as evidence for possible manipulation of the VIX benchmark. Such findings have underscored concerns and contributed to policy discussions around reforms of financial benchmarks (Duffie and Stein, 2015; Gellasch and Nagy, 2019; IOSCO, 2013; Verstein, 2015).

There exists a significant amount of prior work that focuses on the goal of developing trading strategies using reinforcement learning (RL). Previous studies address this in agent-based simulation and with historical data. Numerous simulation-based studies demonstrate the learning of profitable trading studies from a discrete or continuous observation space and an action space (Amrouni et al., 2021; Rummery and Niranjan, 1994; Schvartzman and Wellman, 2009; Sherstov and Stone, 2004; Wright and Wellman, 2018). Likewise many have demonstrated successful RL of trading strategies using historical data. Most employ DRL with a discrete or continuous observation space and discrete action space (Deng et al., 2017; Li et al., 2019; Moody et al., 1998; Nan et al., 2020; Nevmyvaka et al., 2006; Théate and Ernst, 2020; Wu et al., 2020; Zhang et al., 2020), but some recent work considers continuous observation and action spaces (Liu et al., 2020; Ponomarev et al., 2019; Wu et al., 2020; Xiong et al., 2018; Yang et al., 2020). Not surprisingly, given the profit potential of any advantage in trading strategy, advances in RL and DRL are quickly implemented in this domain. What is reported in public research is undoubtedly just the tip of an iceberg.

3. Market Environment

Our model comprises a single security traded through a limit order book, with a transaction-based benchmark calculated at the end of the trading period. This model is implemented in `market-sim`, a market simulation platform originally developed by Wah (2016) and employed in many agent-based finance studies (Wright and Wellman, 2018; Wah et al., 2017; Wang et al., 2021).

3.1. Benchmark

The benchmark we employ is *volume-weighted average price* (VWAP), which Duffie and Dworczak (2018) showed should be hardest to manipulate among a class of transaction-based benchmarks. As its name suggests, VWAP sums the prices weighted by quantity of transactions. Suppose there are N transactions at quantity and price (q_i, p_i) over the trading horizon T. Then VWAP is given by:

$$\beta_T = \frac{\sum_{i=1}^{N} q_i p_i}{\sum_{i=1}^{N} q_i}.$$

In our market scenario, agents submit only single-unit orders, thus the VWAP benchmark becomes:

$$\beta_T = \frac{\sum_{i=1}^{N} p_i}{N}.$$

3.2. Agents in the Market

The benchmark manipulator operates in a market populated by background agents employing the *zero intelligence* (ZI) trading strategy (Gode and Sunder, 1993) in a version described by Brinkman (2018). ZI background traders arrive according to a Poisson process, and on each arrival perceive the market state (current price quote, recent transaction prices, plus a noisy observation of the fundamental),[1] and submit a buy or sell order (decided by coin-flip) for a single unit. The new order replaces its previous offer, if any, on the order book. The price of the limit order at time t is set at the agent's estimated valuation for the good, $v(t)$, offset by a requested surplus ζ_t. Valuation $v(t)$ is the sum of the security's common fundamental value, and an agent-specific private value. Private values are vectors expressing diminishing marginal value for units of the security, drawn i.i.d. from a specified distribution for each agent at the start of the market. The requested surplus ζ_t is chosen for each order

[1] Trading based on a combination of market information and noisy fundamental information is a common feature in agent-based finance studies (Bloembergen et al., 2015; Shearer et al., 2021). Trader attention to market information is necessary for the possibility of spoofing (Wang et al., 2021), and may also provide a channel facilitating benchmark manipulation.

uniformly at random, from an interval whose endpoints are parameters of the ZI strategy. The ZI agent employs one additional strategic parameter, $\eta \in [0, 1]$, in deciding to submit an executable order instead if it would be able to obtain at least fraction η of its requested surplus from the current order book.

Some market instances also include a *market maker* (MM), which follows the MM strategy described by Wah et al. (2017).

4. Benchmark Manipulation Strategies

Like the background traders, the benchmark manipulator operates in the market by submitting single-unit limit orders to buy and sell the market security. Also like these traders, the manipulator accrues profit from the market as the sum of trading cash flow and value of terminal holdings, where this value in turn is the sum of common and private value elements. What distinguishes the manipulator is that it also obtains payoff based on holdings of a contract tied to the benchmark. An example of a manipulator's *contract holdings* may be the stake they have in a publicly traded company the manipulator is trying to sell. If the contract for the manipulator to sell the company is tied to the value of the stock, then the manipulator may benefit by trying to buy shares in the stock market to increase the value of the contract. The benchmark in this example is the trade price of the stock, and the contract holdings is the coefficient of the benchmark to find the manipulator's final payoff from the contract. Specifically in our model, an agent with ψ units of contract holdings receives a payment of $\psi\beta_T$ when the market ends with final benchmark value β_T.

Let $V(t)$ denote the value of the agent's market position at time t, defined as valuation of current market holdings plus cash flow from transactions to that time. The total profit of a benchmark manipulator $B(t)$ is:

$$B(t) = V(t) + \psi\beta_t. \tag{1}$$

If $sign(\psi)$ is positive (negative), then the manipulator benefits from higher (lower) benchmark levels. By choosing higher or lower order prices than it would otherwise, it may be able to influence the benchmark in the direction it would benefit. Doing so entails some loss of profit in the securities market, but may be worthwhile if the gain in payment from contract holdings is sufficient.

4.1. Zero Intelligence Manipulation

The first manipulation strategy we consider is *ZIM*, an adjusted version of the ZI strategy that attempts to influence the benchmark. A standard ZI agent arriving at time t submits orders priced at $p^{\mathbf{ZI}}(t) = v(t) \pm \zeta_t$, where ζ_t is the requested surplus. A ZIM agent offsets $p^{\mathbf{ZI}}(t)$ by $sign(\psi)\chi$, where χ is a strategic parameter:

$$p^{\mathbf{ZIM}}(t) = p^{\mathbf{ZI}}(t) + sign(\psi)\chi. \tag{2}$$

This manipulator also employs the strategic parameter $\eta \in [0, 1]$ to submit a marketable order if the current quote is sufficiently favorable. However, there is a subtle difference in how

η applies for ZIM compared to ZI. For a ZI agent, it is always the case that requested surplus $\zeta_t \geq 0$. However, the offset requested surplus for a ZIM agent may be negative in some cases. If the ZIM agent's total requested surplus $\zeta_t \pm sign(\psi)\chi < 0$, then the manipulator is willing to accept any portion of its requested surplus, rather than just a fraction it like the ZI agent. If buying, the manipulator prices its order at the best available sell order ASK_t rather than $p^{\mathbf{ZIM}}(t)$ if:

$$\text{ASK}_t \leq v(t) + \max\Big\{\eta\big(sign(\psi)\chi - \zeta_t\big), \big(sign(\psi)\chi - \zeta_t\big)\Big\}.$$

If selling, it prices its order at the best available buy order BID_t rather than $p^{\mathbf{ZIM}}(t)$ if:

$$\text{BID}_t \geq v(t) + \min\Big\{\eta\big(sign(\psi)\chi + \zeta_t\big), \big(sign(\psi)\chi + \zeta_t\big)\Big\}.$$

4.2. Manipulation with Deep Reinforcement Learning

We also develop manipulative strategies using DRL. We applied the DRL algorithms deep Q-network (DQN) (Mnih et al., 2015) and deep deterministic policy gradient (DDPG) (Lillicrap et al., 2016) to learn trading strategies that maximize combined payoff from the market and benchmark. We refer to the trading strategies learned through DQN and DDPG as the *DQN agent* and *DDPG agent*, respectively.

Deep Q-Network DQN is a model-free, off-policy value learning algorithm. *Value learning* is the task of inducing a function representing the value of relevant situations. DQN is *model-free* as it does not incorporate explicit representations of the environment dynamics in value learning. A *policy* defines the agent's behavior and is a mapping from states to actions. In the value-based approach, the learned policy is implicit in the learned value function. DQN is *off-policy* as the learned policy may be unconnected from the policy used to generate experiences. Off-policy learning is imperative in our context, as the interface to `market-sim` does not permit updating the policy while the market is active. Thus all training occurs between market runs.

DQN combines Q-learning and deep neural networks (DNNs) to learn Q-values in environments with rich sensory data. A *Q-value* is the estimated value of total discounted reward for the remainder of an episode, for a given state-action pair (s, a). Suppose the agent arrives to the market in state s and takes action a, leading to state s' and producing immediate reward ρ. We record the experience tuple (s, a, s', ρ) to learn from and update Q-values once the episode is complete. DQN uses a DNN to learn a hierarchical abstract representation of a complex state space. This DNN estimates Q-values over a discrete action space. DQN updates the DNN parameters θ using the stochastic gradient descent updating rule:

$$\Delta\theta = \alpha\left[(\rho + \gamma \max_{a'} Q_\theta(s', a')) - Q_\theta(s, a)\right] \nabla_\theta Q_\theta(s, a),$$

where $Q_\theta(s, a)$ is the estimated Q-value given the current DNN parameters and state-action pair, α is the learning rate, and γ is the discount factor.

Deep Deterministic Policy Gradient DDPG is a model-free, off-policy actor-critic algorithm. An *actor-critic* algorithm combines policy learning and value learning. *Policy learning* tries to directly learn a policy function that maximizes the agent's reward. The actor maintains a parametrized policy function and the critic a value function, represented as a DNN (like DQN). The actor is updated given the learned parameters from the critic θ^Q, and by applying the chain rule to the expected return from the distribution J with respect to the parameters of the actor θ^μ:

$$\begin{aligned}\Delta_{\theta^\mu} J &\approx \mathrm{E}_{s_t \sim \nu^\pi}\left[\Delta_{\theta^\mu} Q(s, a \mid \theta^Q)\mid_{s=s_t, a=\mu(s_t \mid \theta^\mu)}\right] \\ &= \mathrm{E}_{s_t \sim \nu^\pi}\left[\Delta_a Q(s, a \mid \theta^Q)\mid_{s=s_t, a=\mu(s_t)} \Delta_{\theta^\mu}\mu(s \mid \theta^\mu)\mid_{s=s_t}\right],\end{aligned}$$

where ν^π is the discounted state visitation distribution for a stochastic behavior policy π. The actor learns a distribution over the action space, which is mapped to a continuous action space. Noise $\mathcal{N}$ is added to the actor's policy for exploration:

$$\mu'(s_t) = \mu(s_t \mid \theta_t^\mu) + \mathcal{N}.$$

State Space The benchmark manipulator's state space includes all the agent's private information. This includes its private valuation of the traded security, contract holdings, and current holdings of the security. We also include the *side* of the current order (buy or sell).

The agent's state space also includes publicly available information in the market, such as the remaining time in the trading period and time since the last trade. We also include features from the market's order book, such as size, spread, and currently listed order prices. The state must be a constant size, but the order book is dynamic throughout the trading period. We address this problem by specifying a limited fixed-size depth of book. We pad or truncate the fixed-depth order book as necessary. For padding, we set prices at estimated final fundamental, plus or minus three standard deviations of the observation noise.

We also include the omega ratio, a metric that determines the favorability of submitting an order. Lastly, we include the number of transactions and their prices. We pad or truncate the transaction price history as necessary to fit the fixed length, similar to our handling of the order book.

Appendix A provides a table of detailed descriptions of the features included in the agent's state space.

Action Space The benchmark manipulator's learned policy selects the price of the order to submit. Upon each arrival, the manipulator perceives the observable state and submits an order. It determines whether to buy or sell by flipping a coin, then submits a single-unit orders for the selected side. There is no option to refrain from submitting an order, but the same effect can be achieved by submitting a noncompetitive orders, far from the current price quotes.

For the DQN agent, the action space is a discrete set of ZIM strategies, each defined by a setting of the ZI parameters plus the offset parameter χ. The set of such ZIM strategies to choose from is specified by the agent designer. On each market arrival, the agent observes state s and evaluates the available actions using the DNN representation of the Q-function. The optimal action $a^* = \arg\max_a Q_\theta(s,a)$—one of the available ZIM strategies—is selected, and applied to the current market state to generate an order for the market.

When the benchmark manipulator uses a policy learned through DDPG to select an action, it directly selects a value $A \in [0,1]$. Our agent then maps this action to a price at time t:

$$p_t^{\mathbf{DDPG}} = \hat{r}_t + \big(C - sign(\psi)\chi\big)A, \tag{3}$$

Where C is some constant treated as a hyperparameter during training, $sign(\psi)$ is the direction of the agent's contract holdings, and χ is an offset parameter. This mapping function is very similar to (2), though rather than randomly selecting a requested profit from a uniform distribution, the agent learns the requested profit directly. After the agent calculates the price, it submits its order containing the price, side, and size.

Reward Function The benchmark manipulator designs its reward function to maximize its profit B from both the market and benchmark (1). We define the agent's reward for the action taken at time t as the difference between the total profit at its next arrival at time t' and the total profit at time t:

$$\rho_t = B(t') - B(t).$$

We capture the total realized profit from the agent's action at time t by calculating the reward as the difference between the total profit at the next arrival and current arrival. This reward cannot be immediately calculated, since the order placed at time t can match with another anytime between t and t' (it is replaced by a new order at t'). Thus, we wait until t' to calculate the reward for the action at time t.

At the end of the market at time T, the summation of the rewards is equivalent to the manipulator's final payoff: market and benchmark:

$$B(T) = \sum_{t \in Arr} \rho_t, \text{ where } Arr \text{ denotes the agent's market arrivals.}$$

5. Experiments

We test the efficacy and implications of benchmark manipulation strategies through agent-based simulation, employing a simplified form of empirical game-theoretic analysis (EGTA) (Tuyls et al., 2020; Wellman, 2016) to identify approximate equilibria among the available strategies. The first question is to what extent agents employing benchmark manipulation strategies—hand-crafted or learned—can influence the benchmark to enhance profit. The second is what are the ramifications for market performance and agent welfare. We evaluate these questions in multiple market environments, employing a variety of strategies for the background agents and benchmark manipulator. In each case, we find the combination of strategies that background traders play in equilibrium in the presence or absence of manipulation. We then evaluate the outcomes in each case, from the perspectives of the manipulator, background agents, and aggregate market.

Table 1: Strategies employed by the background traders (**ZI**).

Strategy	$R_{\min}$	$R_{\max}$	η
$\mathbf{ZI_1}$	0	450	0.5
$\mathbf{ZI_2}$	0	600	0.5
$\mathbf{ZI_3}$	90	110	0.5
$\mathbf{ZI_4}$	140	210	0.5
$\mathbf{ZI_5}$	190	210	0.5
$\mathbf{ZI_6}$	280	380	0.5
$\mathbf{ZI_7}$	380	420	0.5
$\mathbf{ZI_8}$	380	420	1.0
$\mathbf{ZI_9}$	460	540	0.5
$\mathbf{ZI_{10}}$	950	1050	0.5

5.1. Market Environment Settings

Our test environments have fifteen background agents and one benchmark manipulator. The market settings are the same as employed by Wright and Wellman (2018). The market fundamental time series has mean $\bar{r} = 10^5$, mean reversion $\kappa = 0.01$, and market shock variance $\sigma_s = 2 \times 10^4$. The maximum number of units all agents can hold at any time is $q_{\max} = 10$. Lastly, the private value variance is $\sigma^2_{PV} = 2 \times 10^7$. The finite time horizon of the market is $T = 2,000$ time steps. The background agents and manipulator arrive to the market according to a Poisson distribution with rate $\lambda_a = 0.012$.

We consider instances of this market with and without a market maker. If the MM is present, its arrival rate is $\lambda_{mm} = 0.05$. The market maker submits 100 buy orders and 100 sell orders at each market arrival. The spread the market maker uses is 1024 and each order is spaced by 100. The market maker is not considered a player in the market game as its parameters are fixed.

Table 1 specifies the strategies used by the background traders, which are the same as those employed by Wright and Wellman (2018). We use the pure-strategy equilibrium among background traders found by these authors as the baseline no-manipulation case.

In each environment the benchmark manipulator is assigned contract holdings $\psi = 40$. The ZIM agent chooses between the $\mathbf{ZI}_7$ and $\mathbf{ZI}_8$ strategies with possible offsets $\chi \in \{0, 250, 500, 750\}$. Selecting $\chi = 0$ is tantamount to not manipulating. We also examine environments where the manipulator learns trading strategies with DQN or DDPG, using the methods described in Section 4.2. Appendix B presents the hyperparameters selected for DQN and DDPG.

5.2. Simplified EGTA Process

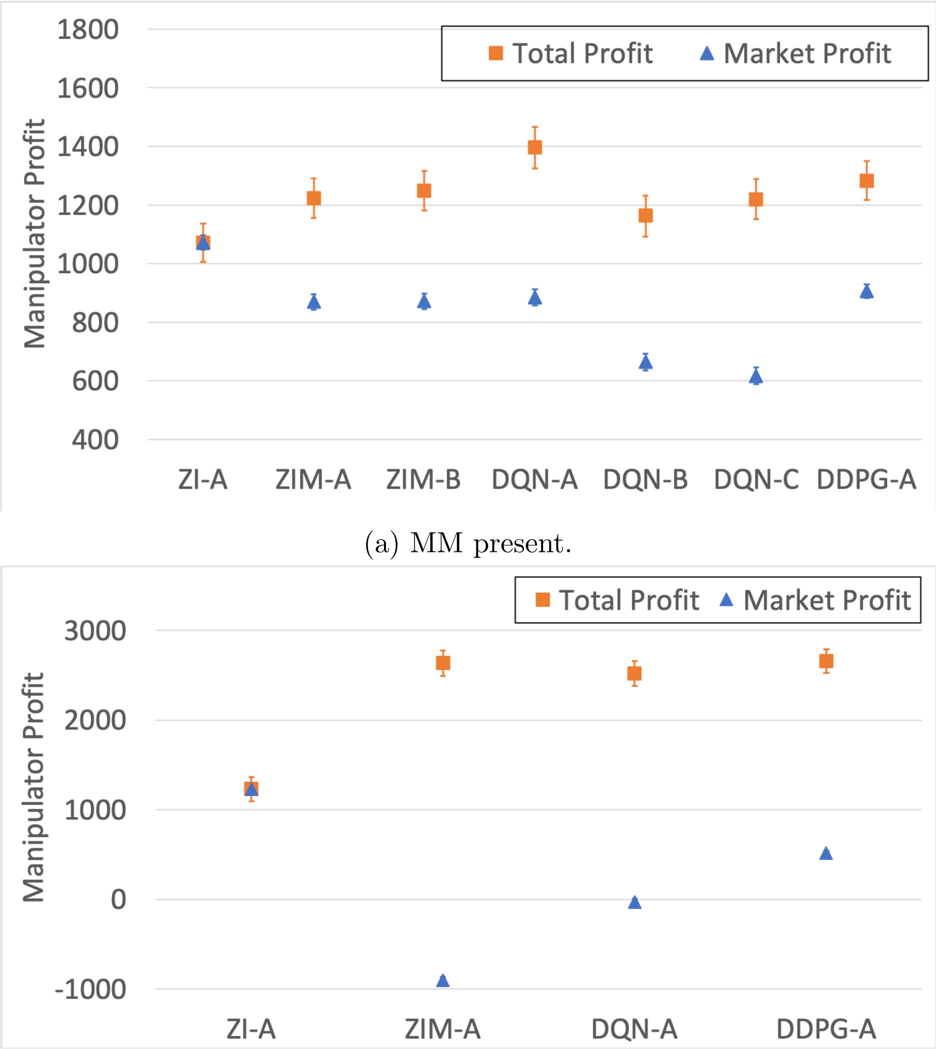
We model the market as a role-symmetric game and partition the agents into two roles: background traders and a single benchmark manipulator. Starting with the baseline no-manipulation equilibrium identified by Wright and Wellman (2018), we replace one of the background traders with a manipulator: implemented as a ZIM, DQN, or DDPG agent. For the ZIM agent, we try each ZIM candidate against the baseline equilibrium and select the most profitable. For the DRL (DQN or DDPG) agents, we likewise train in the context of this baseline.

Once the benchmark manipulation strategy is selected, it is likely that the background traders are no longer in equilibrium. Therefore, we test single-player strategy deviations of the background traders, holding the manipulator strategy fixed. If there is a beneficial single-player deviation, we test a variety of mixed strategies containing the original equilibrium strategy and the strategy of the best deviation s_2. If the original equilibrated strategy is a pure strategy s_1 then this mixed strategy exploration may include the mixture $\Pr(s_1) = \Pr(s_2) = 0.5$. If the background traders deviated to an alternative strategy, we repeat the manipulator strategy optimization (enumerated selection for ZIM, or retraining for the DRL agents). We then repeat the process with another single-player deviation for background traders. If the original equilibrium strategy was pure, the background agents now explore deviating to mixed strategies distributed over three strategies. If the background traders again deviated to another strategy profile, we once again repeat the manipulator strategy optimization. A detailed description of the strategy profiles chosen during this process are presented in Appendix C.

6. Results

We analyze the performance of ZI, ZIM, DQN, and DDPG manipulators. Environment **A** is the market environment where the background agents are equilibrated for no manipulation in a pure strategy equilibrium found by Wright and Wellman (2018). Environment **B** refers to the market environment where the background agents are calibrated to the ZIM manipulator using the single-player deviation method. Environment **C** denotes the market environment where the background agents are calibrated to the DQN manipulator using the single-player deviation method. We include the label “ZI” to signify the case when the agent does not manipulate. We study the welfare impacts of the three manipulators by examining agent and aggregate market payoffs. Specifically, we calculate the market profit and total profit of the benchmark manipulator where total profit aggregates the profit from market trading (i.e., market profit) and profit from the benchmark holdings. We also find the profit of the background traders. The total profit and market profit are the same for the background traders because they are indifferent to the final benchmark calculation. Lastly, we study the aggregate market profit and aggregate total profit. The aggregate market profit is the summation of the background traders’ profit and the benchmark manipulator’s market profit. The aggregate total profit of the system, which we define as the summation of the background traders’ profit and the benchmark manipulator’s the total profit. If a MM is present, then its profit is also included in the aggregate total and market profits.

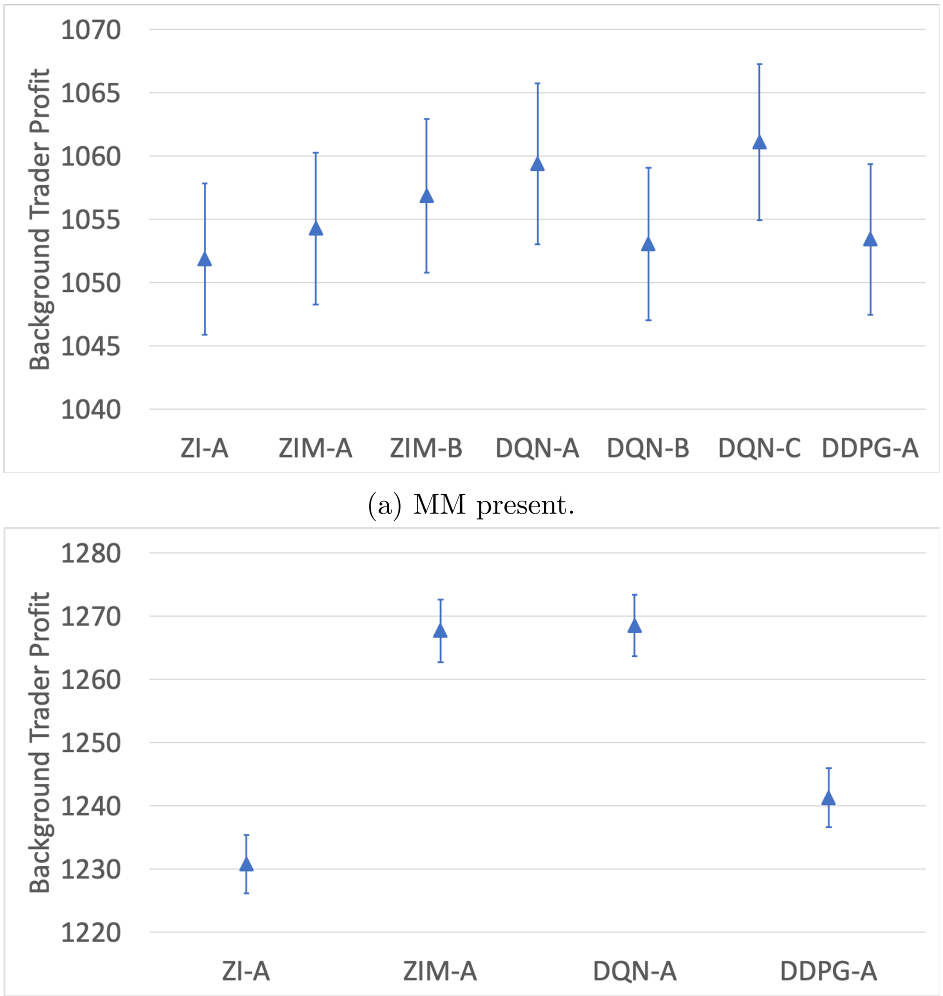
Fig. 1 depicts the total profit and market profit of the benchmark manipulator, respectively. In most cases, the total profit of the benchmark manipulator increases when it



(a) MM present.

(b) No MM present.

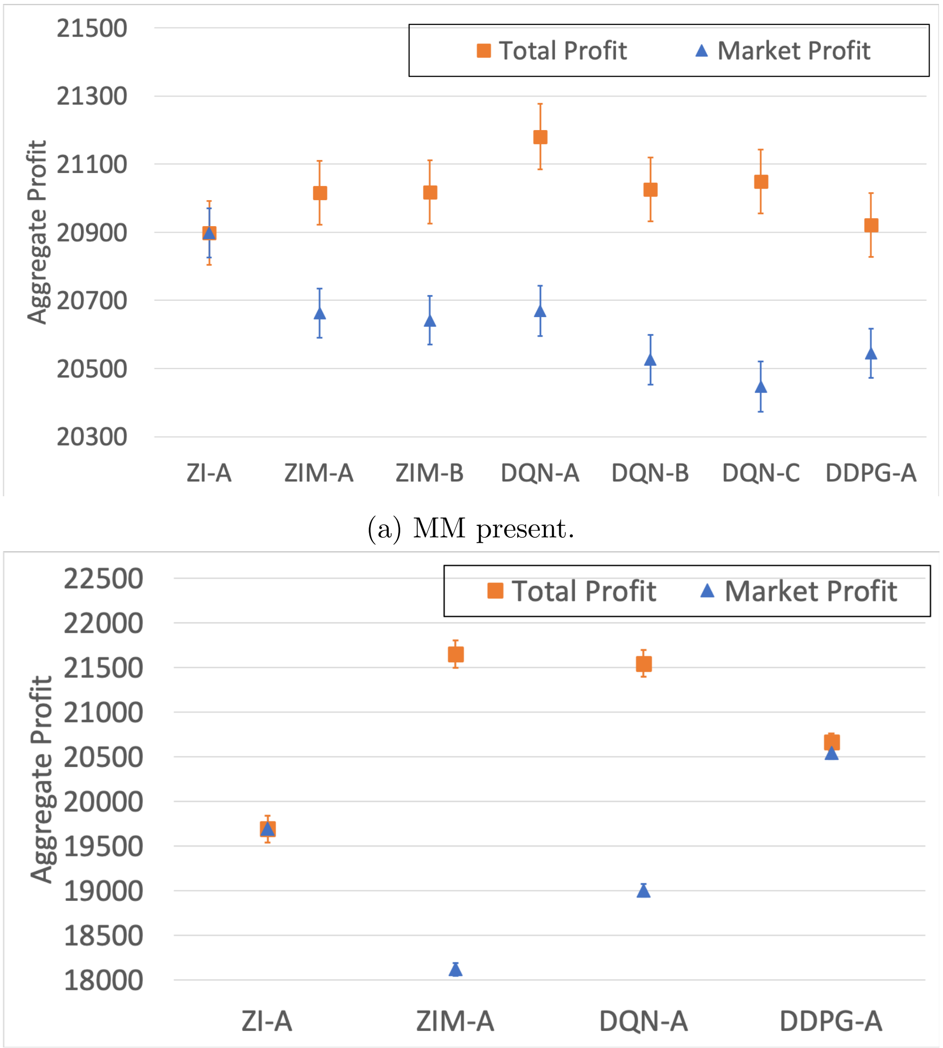
Fig. 1. Profit of the manipulator. In both figures, the x-axis represents which strategy the manipulator uses and in which environment. Each point shows the average payoff of the manipulator with standard error bars.



(a) MM present.

(b) No MM present.

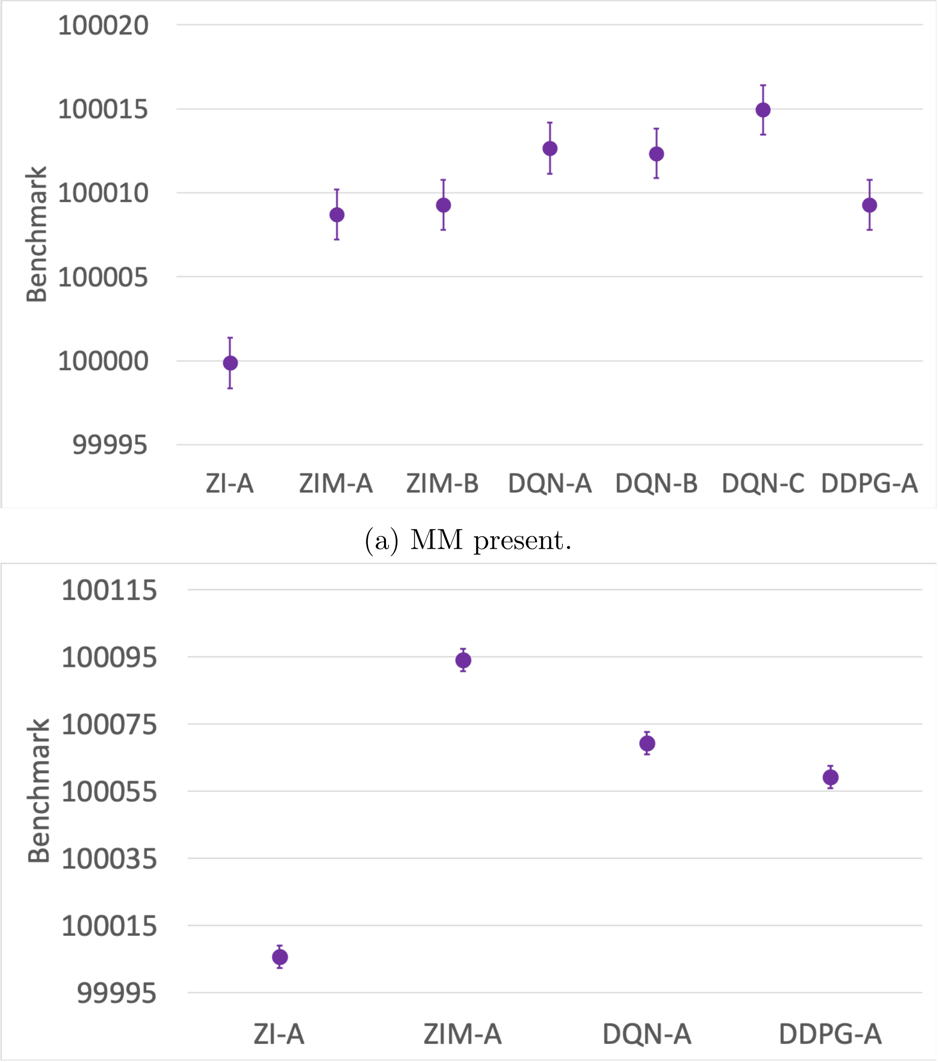
Fig. 2. The aggregate total profit of the fifteen background agents deploying a ZI strategy. The total and market profit is the same for background agents. The x-axis represents which strategy the manipulator uses and in which environment. Each point shows the average aggregate background trader payoff with standard error bars.



(a) MM present.

(b) No MM present.

Fig. 3. Aggregate total and market profit of all agents. In both figures, the x-axis represents which strategy the manipulator uses and in which environment. Each point shows the average aggregate payoff with standard error bars.



(a) MM present.

(b) No MM present.

Fig. 4. The VWAP benchmark under different manipulation strategies and environments. Each point shows the average VWAP with standard error bars.

manipulates the benchmark. When a MM is present, the ZIM agent and DQN agent in **B** and **C** increased their average total profit from the non-manipulative case, but not by a significant amount. These manipulators' market profit decreases from the non-manipulative case. It is worthwhile for the successful manipulator to endure the decrease in market profit because its profits from the change in benchmark more than cover the loss. Also when a MM is present, the DQN agent in Environment **A** and DDPG agent successfully increases its total profit and is the only strategy to do so by a significant amount compared to the non-manipulative agent. When a MM is not present all of the manipulators significantly increase their total profit and decrease their market profit. It is easier for the manipulator to increase its total profit when there is no MM because it does not need to trade through the MM's many orders in the book to change the price.

Fig. 2 shows the profit of the background agents. The background agents benefit from benchmark manipulation as the average payoff of the background agents increases for each manipulative agent compared to the non-manipulative agent. The background agents are significantly better off when there is no MM; this is likely because the manipulator trades with the background agents more in that case. The manipulator's orders are priced to influence the benchmark, which tends to divorce them from market values and in many cases make them more attractive to background traders. The poorly priced orders lead to an increased number of trades. Background traders benefit from the manipulative activity, both from the increase in profitable trades, and from the opportunity to demand higher surplus on orders that would have traded anyway.

Fig. 3 shows the aggregate total profit and aggregate market profit, respectively. The aggregated total profit, which we find by summing the total profit of the benchmark manipulator and background traders. The aggregate total profit increases with benchmark manipulation. The aggregated market profit we find by summing the market profit of the benchmark manipulator and background traders. The aggregate market payoff decreases with market manipulation. The market becomes less efficient when the manipulator is more successful. Benchmark manipulation impacts the benchmark enough that the manipulator's gain from the benchmark exceeds its losses from trading in the market. The background traders gain at most the manipulator's loss from the market, but the manipulator's resulting gain from the external contract exceeds that of the background traders. Therefore, the counterparty to the manipulator in the benchmark contract loses precisely what the benchmark manipulator gains from the benchmark contract.

Fig. 4 depicts the VWAP benchmark in each market environment. The benchmark increases by a significant amount when there is manipulation compared to when there is no manipulation. The benchmark increases more when there is no MM present, though the manipulator is still able to successfully influence the benchmark in both cases. Therefore, the manipulator is able to successfully shift the benchmark in the direction of its contract holdings.

7. Policy Analysis

Following the LIBOR scandal, regulators investigated other benchmarks that had allegedly been manipulated and imposed some of the largest penalties ever paid by financial institutions. Given the important role of benchmarks as financial infrastructure, regulators also turned to potential policy measures to avoid manipulation. The International Organization of Securities Commissions published its Principles for Financial Benchmarks IOSCO (2013) and the European Union adopted its Benchmarks Regulation. Both documents stress the governance obligations of benchmark administrators, the quality of benchmark data, and most relevantly, robust methodological design of benchmarks. Nonetheless, regulators have neither suggested, nor mandated benchmark design features at a microstructure level of granularity. International regulators' interest in developing best practices for benchmark methodology means there should be substantial interest in results along the lines developed here.

The role of regulation is also important because we should expect markets to fail to produce optimal benchmarks themselves. In general, index providers do not operate in fully competitive markets or internalize the full costs and benefits of the indices they produce. There are several reasons for this fact. First, indices are subject to network effects that can cause them to gain a significant degree of lock-in, giving the index provider market power. Second, many widely used benchmarks are produced as a side-effect of other financial activity and do not provide their administrators with a robust revenue stream, notwithstanding that the benchmark can have significant effects on the welfare of counterparties Rauterberg and Verstein (2013). To illustrate, LIBOR originally arose to serve as a reference rate for banks' own lending activities, but came to play a pivotal role in the enormous interest rate derivatives market, without generating any direct revenue for the LIBOR panel banks. As a result of these forces, administrators' private incentives to ensure optimal benchmark design are frequently weaker than what would be socially desirable.

8. Conclusion

We analyze the effectiveness and impact of financial benchmark manipulation, in a simulated market with a single traded security. The manipulator's objective is to shift the benchmark up or down, in order to profit from holdings of a contract tied to the benchmark. The benchmark is transaction-based (VWAP in this study), so potentially influenced by market actions. These actions are costly in that they entail reduced profits or even losses in the primary market. We design and implement three types of benchmark manipulator: one simple hand-crafted strategy, and two derived using deep reinforcement learning.

We find that all three strategies succeed in profitable benchmark manipulation. Presence of a market maker makes manipulation more difficult, and reduces but does not eliminate the manipulative effect. With or without MM, the manipulative activity increases profits of background traders, who thus have no incentives to help mitigate this type of manipulation. Though the aggregate *total* profit of the market participants increases when the benchmark is manipulated, the aggregate *market* profit decreases. As the profit of all market participants increases, it is the non-market counterparties to the benchmark contracts who bear the burden of the manipulation costs. All of these results hold consistently across a range of experimental market environments.

The DRL agents (DQN and DDPG) effectively learn to manipulate, even though they are not given direct instructions to manipulate, or objectives with explicit reference to manipulation. The manipulative strategies emerge naturally from the selection of standard market actions to maximize profits. This learning takes place in an environment of other rationally derived trading strategies, and subjected to adjustment based on presence of the manipulator. To our knowledge, this is the first such demonstration of automated learning of market manipulation strategies.

The apparent ease of learning to manipulate presents serious challenges for financial market regulation. Current manipulation law in the US stock market requires establishment of intent to manipulate, which is arguably not present in this scenario. Given the growing accessibility of DRL technique, it may be worth revisiting these laws to address what might be a seen as a "machine learning loophole" for manipulation (Azzutti et al., 2021).

This study is limited in considering only one particular benchmark, VWAP. Although prior work suggests VWAP is especially robust to market-based manipulation (Duffie and Dworczak, 2018), it may be worthwhile to explore a broader variety of benchmark contexts, including those that employ more complex calculations or are derived from market activity in a qualitatively different way.

Another limitation is that we explore only a short sequence of single-player deviations, rather than using full-blown EGTA to find equilibria. Following a standard iterative EGTA approach like PSRO (Lanctot et al., 2017), we would start from small initial strategy sets for background agents and manipulator, and estimate an empirical game model by simulating combinations of strategies from these sets. At each iteration we extend the sets by computing a best-response to a solution of the current game model. For DRL manipulation, the best response is exactly how we perform strategy generation in this study, training the manipulator against a fixed profile of background traders. We could similarly generate ZI or ZIM agents through a best-response search to expand those sets.

In our current study, we observe that even short sequence of responses produces a relatively stable strategic configuration. We thus consider our results sufficient to demonstrate the essential feasibility of learning to manipulate a transaction-based benchmark, and to support our basic evaluation of the impact of benchmark manipulation on the market.

References

Amrouni, S., Moulin, A., Vann, J., Vyetrenko, S., Balch, T., Veloso, M., 2021. ABIDES-Gym: Gym environments for multi-agent discrete event simulation and application to financial markets. In: *3rd ACM International Conference on AI in Finance*.

Azzutti, A., Ringe, W.-G., Stiehl, H. S., 2021. Machine learning, market manipulation, and collusion on capital markets: Why the "black box" matters. University of Pennsylvania Journal of International Law 43, 79–135.

Banerji, G., 2018. Regulator looks into alleged manipulation of VIX, Wall Street's 'fear index'. Wall Street Journal .

Bariviera, A. F., Guercio, B., Martinez, L. B., Rosso, O. A., 2016. Libor at crossroads: Stochastic switching detection using information theory quantifiers. Chaos, Solitons & Fractals 88, 172–182.

Bloembergen, D., Hennes, D., McBurney, P., Tuyls, K., 2015. Trading in markets with noisy information: An evolutionary analysis. Connection Science 27, 253–268.

Bond, P., Edmans, A., Goldstein, I., 2012. The real effects of financial markets. Annual Review of Financial Economics 4, 339–360.

Boyle, C., 2014. Forex manipulation: How it worked. CNBC.

Brinkman, E., 2018. Understanding Financial Market Behavior through Empirical Game-Theoretic Analysis. Ph.D. thesis, University of Michigan.

Deng, Y., Bao, F., Kong, Y., Ren, Z., Dai, Q., 2017. Deep direct reinforcement learning for financial signal representation and trading. IEEE Transactions on Neural Networks and Learning Systems 28, 653–664.

Duffie, D., 2018. Compression auctions with an application to LIBOR-SOFR swap conversion. Working Paper 3727, Stanford Graduate School of Business.

Duffie, D., Dworczak, P., 2018. Robust benchmark design. Working Paper 3175, Stanford Graduate School of Business.

Duffie, D., Stein, J. C., 2015. Reforming LIBOR and other financial benchmarks. Journal of Economic Perspectives 29, 191–212.

Eisl, A., Jankowitsch, R., Subrahmanyam, M. G., 2017. The manipulation potential of Libor and Euribor. European Financial Management 23, 604–647.

Gellasch, T., Nagy, C., 2019. Benchmark-linked investments: Managing risks and conflicts of interest. Healthy Markets Association.

Gode, D. K., Sunder, S., 1993. Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. Journal of Political Economy 101, 119–137.

Griffin, J. M., Shams, A., 2018. Manipulation in the VIX? Review of Financial Studies 31, 1377–1417.

ICE Benchmark Administration Limited, 2019. U.S. Dollar ICE Bank Yield Index. Intercontinental Exchange.

IOSCO, 2013. Principles for financial benchmarks. The Board of the International Organization of Securities Commissions.

Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Pérolat, J., Silver, D., Graepel, T., 2017. A unified game-theoretic approach to multiagent reinforcement learning. In: *31st Annual Conference on Neural Information Processing Systems*, pp. 4190–4203.

Li, Y., Zheng, W., Zheng, Z., 2019. Deep robust reinforcement learning for practical algorithmic trading. IEEE Access 7, 108014–108022.

Lillicrap, T., Hunt, J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D., 2016. Continuous control with deep reinforcement learning. In: *4th International Conference on Learning Representations*.

Liu, Y., Liu, Q., Zhao, H., Pan, Z., Liu, C., 2020. Adaptive quantitative trading: An imitative deep reinforcement learning approach. vol. 34, pp. 2128–2135.

Martínez-Miranda, E., McBurney, P., Howard, M. J. W., 2016. Learning unfair trading: A market manipulation analysis from the reinforcement learning perspective. In: *IEEE Conference on Evolving and Adaptive Intelligent Systems*, pp. 103–109.

McBride, J., 2016. Understanding the Libor Scandal. Council on Foreign Relations.

Mizuta, T., 2020. Can an AI perform market manipulation at its own discretion? A genetic algorithm learns in an artificial market simulation. In: *IEEE Symposium Series on Computational Intelligence*, pp. 407–412.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D., 2015. Human-level control through deep reinforcement learning. Nature 518, 529–533.

Moody, J., Wu, L., Liao, Y., Saffell, M., 1998. Performance functions and reinforcement learning for trading systems and portfolios. Journal of Forecasting 17, 441–470.

Nan, A., Perumal, A., Zaiane, O. R., 2020. Sentiment and knowledge based algorithmic trading with deep reinforcement learning.

Nevmyvaka, Y., Feng, Y., Kearns, M., 2006. Reinforcement learning for optimized trade execution. In: *23rd International Conference on Machine learning*, pp. 673—-680.

Ponomarev, E. S., Oseledets, I. V., Cichocki, A. S., 2019. Using reinforcement learning in the algorithmic trading problem. Journal of Communications Technology and Electronics 64, 1450–1457.

Rauch, B., Goettsche, M., El Mouaaouy, F., 2013. LIBOR manipulation: Empirical analysis of financial market benchmarks using Benford's law. SSRN Electronic Journal .

Rauterberg, G., Verstein, A., 2013. Index theory: The law, promise and failure of financial indices. Yale Journal on Regulation 30, 1.

Rummery, G. A., Niranjan, M., 1994. On-line q-learning using connectionist systems. Tech. rep.

Schvartzman, L. J., Wellman, M. P., 2009. Stronger CDA strategies through empirical game-theoretic analysis and reinforcement learning. In: *8th International Conference on Autonomous Agents and Multiagent Systems*.

Shearer, M., Byrd, D., Balch, T. H., Wellman, M. P., 2021. Stability effects of arbitrage in exchange traded funds: An agent-based model. In: *2nd ACM International Conference on Artificial Intelligence in Finance*.

Sherstov, A. A., Stone, P., 2004. Three automated stock-trading agents: A comparative study. In: *AAMAS-04 Workshop on Agent-Mediated Electronic Commerce*, pp. 173–187.

Théate, T., Ernst, D., 2020. An application of deep reinforcement learning to algorithmic trading.

Tuyls, K., Perolat, J., Lanctot, M., Hughes, E., Everett, R., Leibo, J. Z., Szepesvári, C., Graepel, T., 2020. Bounds and dynamics for empirical game-theoretic analysis 34.

Verstein, A., 2015. Benchmark manipulation. Boston College Law Review 56, 215–272.

Wah, E., 2016. Computational Models of Algorithmic Trading in Financial Markets. Ph.D. thesis, University of Michigan.

Wah, E., Wright, M., Wellman, M. P., 2017. Welfare effects of market making in continuous double auctions. Journal of Artificial Intelligence Research 59, 613–650.

Wang, X., Hoang, C., Vorobeychik, Y., Wellman, M. P., 2021. Spoofing the limit order book: A strategic agent-based analysis. Games 12.

Wellman, M. P., 2016. Putting the agent in agent-based modeling. Autonomous Agents and Multi-Agent Systems 30, 1175–1189.

Wright, M., Wellman, M. P., 2018. Evaluating the stability of non-adaptive trading in continuous double auctions. In: *17th International Conference on Autonomous Agents and Multiagent Systems*.

Wu, X., Chen, H., Wang, J., Troiano, L., Loia, V., Fujita, H., 2020. Adaptive stock trading strategies with deep reinforcement learning methods. Information Sciences 538, 142–158.

Xiong, Z., Liu, X.-Y., Zhong, S., Yang, H., Walid, A., 2018. Practical deep reinforcement learning approach for stock trading.

Yagemann, C., Chung, P. H., Uzun, E., Ragam, S., Saltaformaggio, B., Lee, W., 2021. Modeling large-scale manipulation in open stock markets. IEEE Security and Privacy 19, 58–65.

Yang, H., Liu, X.-Y., Zhong, S., Walid, A., 2020. Deep reinforcement learning for automated stock trading: An ensemble strategy. SSRN Electronic Journal pp. 1–9.

Zhang, Z., Zohren, S., Roberts, S., 2020. Deep reinforcement learning for trading. Journal of Financial Data Science 2, 25–40.

Appendix A. Table of the Benchmark Manipulator's State Space

Table 2: Description of each state space feature utilized by our DQN and DDPG manipulators.

Feature	**Description**
Private bid	The private value of the next unit bought.
Private ask	The private value of the next unit sold.
Market holdings	The agent's current holdings of the traded asset. (If > 0, then bought more units than sold, and if < 0, then sold more units than bought.)
Contract holdings	The agent's holdings from an external contract whose valuation depends on this market. (Multiplied by the direction the agent is more profitable in, i.e. if the agent is better off if the valuation goes up, then this value is positive. If it's better off if the valuation goes down, this value is positive.)
Side	If the agent will submit a buy or sell order (currently all agents flip a coin in market-sim to determine the side).
Final fundamental estimate	An estimate of the final fundamental. (A noisy observation of the mean reverting time series representing the fundamental value of the traded asset.)
Time until end	The number of time steps remaining in the trading period.
Bid omega ratio	Estimates the "favorability" of submitting a buy order at the current time. Ratio of (recent trade prices) higher than (the agent's estimated value of the asset) to (recent trade prices) lower than (the agent's estimated value of the asset). Only considers the last X trades.

Table 2: Description of each state space feature utilized by our DQN and DDPG manipulators.

Feature	**Description**
Ask omega ratio	Estimates the "favorability" of submitting a sell order at the current time. Ratio of (recent trade prices) higher than (the agent's estimated value of the asset) to (recent trade prices) lower than (the agent's estimated value of the asset). Only considers the last X trades.
Bid size	Depth of book, bid. The number of active buy orders in the market.
Ask size	Depth of book, ask. The number of active sell orders in the market.
Spread	The difference in price between the best available ask and the best available bid in the book. $\min(\text{sell price}) - \max(\text{buy price})$
Bid vector	An ordered, padded vector of the difference between the price of all active buy orders and the estimated value. Organized in ascending order by price, then time, i.e. the highest-priced bid is the last element, and if two orders have the same price, then the order that arrived first has the higher index. If there are fewer active buy orders than the length of the vector, then it is padded with very low bid prices.
Ask vector	An ordered, padded vector of the difference between the estimated value and the price of all active sell orders. Organized in descending order by price, then time, i.e. the lowest-priced ask is the first element, and if two orders have the same price, then the order that arrived first has the lower index. If there are fewer active sell orders than the length of the vector, then it is padded with very high sell prices.
Number of transactions	The current number of trades that have occurred

Table 2: Description of each state space feature utilized by our DQN and DDPG manipulators.

Feature	Description
	in the market.
Transaction history	A padded ordered list of the difference between the estimated value and the price trades. Organized in descending order by time, i.e. the most recent trade is the first element. If there are fewer trades than the length of the vector, then it is padded with zeros.

Appendix B. Deep Reinforcement Learning Hyperparameters

Tables 3–6 list the hyperparameters used to train our DQN and DDPG agents when MMs are present and when MMs are not present.

Table 3: The hyperparameters of DQN when a MM is present.

Hyperparameter	Value
Number of episodes	2,500
Batch size	1,024
Replay capacity	20,000
Minimum replay size	2,500
Number of gradient steps per update	5
Target updated period	30
Polyak update	True
Error clipping	100.0
Size of network	26
Learning rate	1e-6
Exploration schedule	Constant = 0.2
Reward clipping	100
Omega depth	5
Length of bid vector	5
Length of ask vector	5
Length of transaction vector	5

Table 4: The hyperparameters of DDPG when a MM is present.

Hyperparameter	Value
Number of episodes	70,000
Batch size	512
Replay capacity	20,000
Minimum replay size	10,000
Number of gradient steps per update	10
Target updated period	30
Target network weight	0.005
Discount factor	0.99
Error clipping	1.0
Reward clipping	10,000
Size of network	128
Learning rate	3e-5
Exploration noise	0.1
Action coefficient C	1,050
Benchmark impact χ	500
Omega depth	5
Length of bid vector	5
Length of ask vector	5
Length of transaction vector	5

Table 5: The hyperparameters of DQN when a MM is **not** present.

Hyperparameter	Value
Number of episodes	2,500
Batch size	1,024
Replay capacity	20,000
Minimum replay size	2,500
Number of gradient steps per update	5
Target updated period	30
Polyak update	False
Error clipping	False
Size of network	26
Learning rate	1e-5
Exploration schedule	Constant = 0.2
Reward clipping	False
Omega depth	5
Length of bid vector	5
Length of ask vector	5
Length of transaction vector	5

Table 6: The hyperparameters of DDPG when a MM is **not** present.

Hyperparameter	Value
Number of episodes	110,000
Batch size	1,024
Replay capacity	20,000
Minimum replay size	2,500
Number of gradient steps per update	10
Target updated period	30
Target network weight	0.005
Discount factor	0.99
Error clipping	1.0
Reward clipping	40,000
Size of network	64
Learning rate	3e-5
Exploration noise	0.3
Action coefficient C	1,050
Benchmark impact χ	1,250
Omega depth	5
Length of bid vector	5
Length of ask vector	5
Length of transaction vector	5

Appendix C. Equilibria and Deviations in Benchmark Manipulation Games

Tables 7–15 show the strategy deviations from the equilibrated game generated by Wright and Wellman (2018).

Table 7: Environment **B** with a ZIM manipulator. Benchmark manipulator deviation.

Payoff	Benchmark Manipulator							
	ZIM$_1$	**ZIM**$_2$	**ZIM**$_3$	**ZIM**$_4$	**ZIM**$_5$	**ZIM**$_6$	**ZIM**$_7$	**ZIM**$_8$
1084.79	1	0	0	0	0	0	0	0
1274.31	0	1	0	0	0	0	0	0
1288.53	0	0	1	0	0	0	0	0
1191.94	0	0	0	1	0	0	0	0
1106.50	0	0	0	0	1	0	0	0
1228.72	0	0	0	0	0	1	0	0
1310.98	0	0	0	0	0	0	1	0
1188.44	0	0	0	0	0	0	0	1

Table 8: Environment **B** with a ZIM manipulator. Single player deviation.

Payoff	Background									
	ZI$_1$	**ZI**$_2$	**ZI**$_3$	**ZI**$_4$	**ZI**$_5$	**ZI**$_6$	**ZI**$_7$	**ZI**$_8$	**ZI**$_9$	**ZI**$_{10}$
1057.98	1	0	0	0	0	0	14	0	0	0
1053.16	0	1	0	0	0	0	14	0	0	0
1061.28	0	0	1	0	0	0	14	0	0	0
1053.04	0	0	0	1	0	0	14	0	0	0
1071.51	0	0	0	0	1	0	14	0	0	0
1061.73	0	0	0	0	0	1	14	0	0	0
1058.02	0	0	0	0	0	0	15	0	0	0
1055.22	0	0	0	0	0	0	14	1	0	0
1054.44	0	0	0	0	0	0	14	0	1	0
1059.99	0	0	0	0	0	0	14	0	0	1

Table 9: Environment **B** with a ZIM manipulator. Mixed player deviation.

Payoff	Background									
	$\mathbf{ZI}_1$	$\mathbf{ZI}_2$	$\mathbf{ZI}_3$	$\mathbf{ZI}_4$	$\mathbf{ZI}_5$	$\mathbf{ZI}_6$	$\mathbf{ZI}_7$	$\mathbf{ZI}_8$	$\mathbf{ZI}_9$	$\mathbf{ZI}_{10}$
1055.32	0	0	0	0	12	0	3	0	0	0
1046.07	0	0	0	0	10	0	5	0	0	0
1043.70	0	0	0	0	8	0	7	0	0	0
1051.70	0	0	0	0	7	0	8	0	0	0
1049.55	0	0	0	0	5	0	10	0	0	0
154.03	0	0	0	0	3	0	12	0	0	0
1071.51	0	0	0	0	1	0	14	0	0	0

Table 10: Environment **B** with a ZIM manipulator. Benchmark manipulator deviation.

Payoff	Benchmark Manipulator							
	$\mathbf{ZIM}_1$	$\mathbf{ZIM}_2$	$\mathbf{ZIM}_3$	$\mathbf{ZIM}_4$	$\mathbf{ZIM}_5$	$\mathbf{ZIM}_6$	$\mathbf{ZIM}_7$	$\mathbf{ZIM}_8$
1030.30	1	0	0	0	0	0	0	0
1179.88	0	1	0	0	0	0	0	0
1302.12	0	0	1	0	0	0	0	0
1065.72	0	0	0	1	0	0	0	0
1065.54	0	0	0	0	1	0	0	0
1275.00	0	0	0	0	0	1	0	0
1191.44	0	0	0	0	0	0	1	0
1222.44	0	0	0	0	0	0	0	1

Table 11: Environment **B** with a ZIM manipulator. Second single player deviation.

Payoff	Background									
	$\mathbf{ZI}_1$	$\mathbf{ZI}_2$	$\mathbf{ZI}_3$	$\mathbf{ZI}_4$	$\mathbf{ZI}_5$	$\mathbf{ZI}_6$	$\mathbf{ZI}_7$	$\mathbf{ZI}_8$	$\mathbf{ZI}_9$	$\mathbf{ZI}_{10}$
1049.47	1	0	0	0	1	0	13	0	0	0
1050.23	0	1	0	0	1	0	13	0	0	0
1055.74	0	0	1	0	1	0	13	0	0	0
1057.71	0	0	0	1	1	0	13	0	0	0
1061.30	0	0	0	0	2	0	13	0	0	0
1066.48	0	0	0	0	1	1	13	0	0	0
1062.47	0	0	0	0	1	0	14	0	0	0
1058.84	0	0	0	0	1	0	13	1	0	0
1053.99	0	0	0	0	1	0	13	0	1	0
1054.64	0	0	0	0	1	0	13	0	0	1

Table 12: Environment **B** with a ZIM manipulator. Three strategy mixed player deviation.

Payoff	Background									
	ZI$_1$	**ZI**$_2$	**ZI**$_3$	**ZI**$_4$	**ZI**$_5$	**ZI**$_6$	**ZI**$_7$	**ZI**$_8$	**ZI**$_9$	**ZI**$_{10}$
1048.60	0	0	0	0	5	5	5	0	0	0
1057.09	0	0	0	0	1	4	10	0	0	0
1048.97	0	0	0	0	4	1	10	0	0	0
1052.14	0	0	0	0	1	2	12	0	0	0
1049.40	0	0	0	0	2	1	12	0	0	0
1066.48	0	0	0	0	1	1	13	0	0	0

Table 13: Environment **B** with a ZIM manipulator. Benchmark manipulator deviation.

Payoff	Benchmark Manipulator							
	ZIM$_1$	**ZIM**$_2$	**ZIM**$_3$	**ZIM**$_4$	**ZIM**$_5$	**ZIM**$_6$	**ZIM**$_7$	**ZIM**$_8$
1096.57	1	0	0	0	0	0	0	0
1235.81	0	1	0	0	0	0	0	0
1330.30	0	0	1	0	0	0	0	0
1212.72	0	0	0	1	0	0	0	0
1120.04	0	0	0	0	1	0	0	0
1117.93	0	0	0	0	0	1	0	0
1225.45	0	0	0	0	0	0	1	0
1132.41	0	0	0	0	0	0	0	1

Table 14: Environment **B** with a DQN manipulator. Single player deviation.

Payoff	Background									
	ZI$_1$	**ZI**$_2$	**ZI**$_3$	**ZI**$_4$	**ZI**$_5$	**ZI**$_6$	**ZI**$_7$	**ZI**$_8$	**ZI**$_9$	**ZI**$_{10}$
1055.01	1	0	0	0	0	0	14	0	0	0
1064.48	0	1	0	0	0	0	14	0	0	0
1053.80	0	0	1	0	0	0	14	0	0	0
1053.39	0	0	0	1	0	0	14	0	0	0
1053.09	0	0	0	0	1	0	14	0	0	0
1059.18	0	0	0	0	0	1	14	0	0	0
1053.44	0	0	0	0	0	0	15	0	0	0
1064.49	0	0	0	0	0	0	14	1	0	0
1064.99	0	0	0	0	0	0	14	0	1	0
1051.96	0	0	0	0	0	0	14	0	0	1

Table 15: Environment **B** with a DQN manipulator. Mixed player deviation.

Payoff	Background									
	$\mathbf{ZI}_1$	$\mathbf{ZI}_2$	$\mathbf{ZI}_3$	$\mathbf{ZI}_4$	$\mathbf{ZI}_5$	$\mathbf{ZI}_6$	$\mathbf{ZI}_7$	$\mathbf{ZI}_8$	$\mathbf{ZI}_9$	$\mathbf{ZI}_{10}$
1058.95	0	0	0	0	0	0	3	0	12	0
1042.39	0	0	0	0	0	0	5	0	10	0
1061.90	0	0	0	0	0	0	7	0	8	0
1054.41	0	0	0	0	0	0	8	0	7	0
1054.43	0	0	0	0	0	0	10	0	5	0
1054.01	0	0	0	0	0	0	12	0	3	0
1064.99	0	0	0	0	0	0	14	0	1	0