Multi-Agent Reinforcement Learning for Assumption-Free Stock Market Modeling

Technical Report

Maverick Zhang
UC Berkeley
Berkeley, CA 94720, United States
`maverickzhang@berkeley.edu`

Juanwu Lu
UC Berkeley
Berkeley, CA 94720, United States
`juanwu_lu@berkeley.edu`

Abstract

Modeling consumer behavior in a market has long been an exciting challenge for economic research. A practical model can help predict and understand market behaviors and prevent fluctuations or unexpected breakdowns. However, existing modeling approaches either rely on restrictive assumptions or require complicated optimization procedures for online approximations. With recent progress in machine learning, deep reinforcement learning give rise to some state-of-the-art trading and forecasting techniques in economics. Nevertheless, more effort needs to focus on modeling the market behavior. This paper seeks to investigate building an assumption-free stock market model that depicts fine-grained individual decisions. A detailed stock market environment is proposed to incorporate multi-agent deep deterministic policy gradients (MADDPG) for modeling behaviors. Results from the experiments show that deterministic policy can result in the agents being trapped at a Nash Equilibrium.

Keywords Stock Market Modeling · Reinforcement Learning · Multi-Agent Deep Deterministic Policy Gradient

1 Introduction

Modeling behavior in a market is critical in economics, for all institutions: firms, households, and governments. Getting the model correct crucially allows preventative measures to be taken for the economy, especially when trying to prevent economic crises such as crashes or recessions. Since 2008, the market fragility hypothesis proposed by Hyman Minsky has been widely adopted Minsky [1992] as an explanation for why such large recessions have become commonplace.

Subject to any shock, such as the COVID pandemic, we see the effects of market fragility such as runaway inflation; hence, it is an exciting challenge to build a model that may provide a more accurate prediction of the economy. To that end, a model should be sensitive and able to reflect individuals' choices with high fidelity and be largely assumption free. Since the development of macroeconomics, economic models have often needed to adopt extreme simplifying assumptions in order to be computationally feasible: leading to a problem in aggregation. Thus, while agent based modelling should be adopted (better predictions, but less deterministic and more computationally complex), most economic models in use for forecasting are based upon deterministic models derived from simplifying assumptions (simpler, deterministic, but are biased). Unfortunately, such models contributed to the 2008 recession, and flawed assumptions has historically prolonged economic disasters such as the Great Depression as well, a la Hooverism.

Thus we seek to take first steps towards bridging this gap and develop a model that is computationally less complex, deterministic, and largely assumption free, taking multi agent reinforcement learning methods as our approach.

2 Related Work

2.1 Aggregation Problem in Economics

The aggregation problem is a well known problem that makes economics modeling hard. Due to complex interactions and emergent behavior, economic systems exhibit behavior that can be different on an aggregate and individual level, such as risk preference [Blackburn and Ukhov, 2008]. To deal with this, the microfoundations are abstracted away and forgotten, which reduces complexity enough to make economic systems modelable. In macroeconomics, the premier model is the Dynamic Stochastic General Equilibrium (DSGE) model [Galí, 2015]. However, abstracting away the microfoundations restricts the modelling of emergent behavior. This can sometimes result in bad modelling predictions that have enormous consequences, such as the 2008 recession.

The opposite approach to this type of modeling is Agent-based Computation Economics (ACE) models [Niazi and Hussain, 2011] which seeks to begin with agents and keep the microfoundations, while being computationally expensive. But the benefit of ACE is that it can with minimal assumptions predict and exhibit emergent behavior.

2.2 Reinforcement Learning for Economic Modelling

Reinforcement learning (RL) is well-known for its ability to learn human-level control through experience-driven training. Figure 1 shows the basic logic that drives a reinforcement learning system. Progress in RL algorithms promotes its applications for modeling and understanding behaviors in complex systems, such as traffic flow [Lussange et al., 2021], human locomotion control [Song et al., 2021], and many others. Existing literature has explored utilizing RL to help with forecasting and trading. Lee applied a temporal-difference algorithm in [Lee, 2001], for daily stock price prediction. Compared with supervised learning, the algorithm is more robust for long-term prediction with delayed rewards. Sornmayura [Sornmayura, 2019] incorporated a DQN model trained on past historical data for foreign exchange price forecasting in its trading system. In [Ertuğrul and Tağluk, 2018], generalized behavioral learning helped better forecast financial indicators on a 3-month basis. For automatic trading, Corazza and Sangalli compared Q-learning and SARSA for their applications in stock trading [Corazza and Sangalli, 2015]. Carapuco et al. [Carapuço et al., 2018] designed a system using classic reinforcement learning for foreign exchange trading. Conegundes et al. [Conegundes and Pereira, 2020] demonstrated the power of Deep RL by beating the stock market in a day trading system. Nevertheless, reinforcement learning has yet to make its way into economics modeling [Tilbury, 2022].

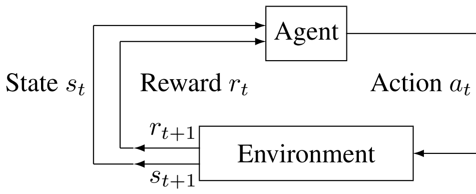

Figure 1: The interaction between agent and environment in reinforcement learning.

Before this work, Lussange et al. [Lussange et al., 2021] explored modeling stock markets with multi-agent reinforcement learning in their paper. However, the model is trained on a daily trading basis, an aggregation over temporal dimension that may fail to capture minor fluctuations in buyers' decisions. Instead, in our model, we allow agents to trade every 5 minutes under clearance, which fits the settings of a real stock market. We will cover more details on our environment in the following section.

3 Methodology

This paper aims to build a stock market model between the DSGE and ACE models, where we incorporate multi-agent deep reinforcement learning to learn the behaviors of agents in a stock market directly via interactions to reduce online optimizations seen in ACE models while not abstracting the microfoundations as seen in DSGE models. In this section, we present detailed settings of the stock market environments, along with the multi-agent deep deterministic policy gradients (MADDPG) we used to learn agents' behaviors. Compared to the classic models, our proposed model can obtain emergent behavior without assumptions and prevent the complexity from blowing up.

3.1 Stock Market Environment

3.1.1 Agents, Market, and Trading System

Instead of modeling a colossal system with millions of agents, our model uses macro agents intended to imitate sectors or clusters of populations in the economy. For instance, populations in real life may be stratified by wealth and the ability to know latent information within a market. Therefore, in our environment, we initialize agents with different initial funding, shareholdings, and masks of available information. These agents are driven by an intuitive objective, maximizing the final amount of money for each environment episode. Each episode of the environment lasts 390 timesteps, corresponding to the total daily trading minutes in a real stock market. Each deal agents make must go through a clearance procedure, where a queue holds and randomly assigns bidders and sellers for buying and selling their holdings. This process is shown in Algorithm 1.

```
Data: Agent price proposals and quantities $(p_1, q_1), \dots, (p_n, q_n)$
Result: Returns $\Delta S$ and $\Delta B$ made per agent
buyers := $\{i \mid q_i > 0\}$;
sellers := $\{i \mid q_i < 0\}$;
$(\Delta S_i, \Delta B_i) = (0, 0)$ for all $i \leq n$;
randomize(buyers);
randomize(sellers);
while buyers $\neq \emptyset$ and sellers $\neq \emptyset$ do
    b $\in$ buyers;
    s $\in$ sellers;
    if $p_b \geq p_s$ then
        $\Delta B_b = \Delta B_b - p_b \cdot q_b$;
        $\Delta S_b = \Delta S_b + q_s$;
        $q_b = q_b - q_s$;
        $\Delta B_s = \Delta B_s + p_s \cdot q_s$;
        $\Delta S_s = \Delta S_s - q_s$;
        $q_s = q_s + q_b$;
    else
        continue;
    end
    if $q_b$ = 0 then
        delete b from buyers;
    end
    if $q_s$ = 0 then
        delete $s$ from buyers;
    end
end
```

Algorithm 1: The clearance procedure

3.1.2 State

Their interactions in the market determine the price of a stock and therefore the worth of their shares held. In addition, to give them information about their fellow competitors in the market, we provide stocks that change based upon the volatility of the agents bids and asks. In addition, we provide dummy stocks that change according to Brownian motion that are independent of the volatility. This way each agent can obtain a very noisy estimate of their opponents. In addition, we mask out some of the stocks depending on the agent. That is, not all agents are focused on all the stocks and may get noisier or less noisy estimates of their competitor's actions.

The state is therefore:

$$s_t = (P_t, \mathbf{c}_t, \mathbf{u}_t, M, B_t, S_t) \tag{1}$$

where P_t is the current price of the stock, $\mathbf{c}_t$ is the vector of correlated stocks, $\mathbf{u}_t$ is the vector of uncorrelated stocks (Brownian motion), M is a mask vector to determine which stocks are being considered by an agent, B_t is the money held by the agent (real assets), and S_t is the shares held by the agent (financial assets)

3.1.3 Action

Agents can either bid or sell an asset, and they can determine the price they are willing to sell. We implement limit transactions and only allow a transaction to proceed if another agent is willing to make the deal. Now the agent therefore has a two dimensional action

$$a_t = (p_t, q_t) \tag{2}$$

where p_t is the price of a bid or ask, and q_t is the quantity of shares to buy (positive) or sell (negative).

3.1.4 Rewards

Their rewards simply their real assets and their financial assets summed. We provide an environment parameter ω that determines the relative importance of shares held vs. money. We also place a decay on the real assets currently held by an agent (γ) so agents are incentivized to enter the market and not hold. In addition, we must have the current stock price in order to convert shares held into a common unit. Thus, our rewards are

$$r(s_t, a_t) := \omega S_t P_t + B_t \tag{3}$$

where the assets at each time step

3.2 Multi-Agent Deep Deterministic Policy Gradient

We use Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [Lowe et al., 2017], a commonly used multi-agent reinforcement learning algorithm to try and learn our economic model. The critical insight of MADDPG is the centralized-training-decentralized execution structure, where agents' Q-functions utilize the global state and actions from other agents to stabilize the training. We argue that this setting holds for real-world situations since many of the trading platforms would allow bidders to view others' bids.

For our implementations, each agent is represented by a policy network $\mu^{\theta}(a_t|s_t)$, and a critic network $Q^{\phi}(s_t, a_t)$. Both are three-layer MLP with 64 neurons each layer. Models are trained on a learning rate of 0.0001 and a reward discount factor of 0.99. Algorithm 2 shows the training procedure of MADDPG.

4 Experiment

4.1 Environment Settings and Training

To facilitate our investigation into agents' trading strategies, we compared results trained with four groups of hyper-parameter and action settings, including different budget discounts (γ), worth of stock (ω), an action space with share quantity, and an action space applied all or nothing rule. As mentioned in 3.1.4, a budget discount is analogous to the inflation rate in real life and acts as incentive for agents to enter the market. Generally, mild inflation can increase stock trading, while hyperinflation decreases it. The worth of stock is a ratio factor, which indicates how much an agent would treasure its stock holdings against its cash. We run our training on two market rules, one with the "all or nothing" rule, which means an agent has to sell all his shares or sell nothing, whereas the other allows agents to decide the quantity to buy or sell. Debts are not allowed in our environment, meaning any agent with its current amount of cash below 0 would terminate the episode. Results from all our experiments are shown in Figures 2-5.

Training curves from all our experiments are shown in Figures 2-5. Overall, the curves demonstrate similar patterns, where agents start with competing and aggressive strategies, causing episodic return oscillations. By the end of the training, agents from the same experiment settings will have their episode total returns stabilized at approximately the same level. The difference in episodic returns' scale is affected by random initialization and budget discount factors.

The convergence of episodic total returns can usually be associated with equilibrium, where agents consent to take conservative actions for either selfish reasons or the public good. We further investigate the potential causes for the convergence in our results and explain from the perspective of Nash Equilibrium in the following sections.

4.2 Underestimation of Q Values

Given that all the agents, in both problem settings and across a wide range of hyperparameters converge upon the same strategy: hold their shares, this suggests some system wide issue prevalent within our approach. Figure 6 shows the Q-values against the actual returns (dashed lines). We find that there is a systematic underestimation of Q-values, with the actual return far higher

```
Data: Batch size $S$; Reward discount $\gamma_r$; Replay buffer $\mathcal{D}$; Number of agents $N$; Total number of episodes
      $M$; Random process $\mathcal{N}$ for exploration; Episode index $i = 0$.
while $i < M$ do
    Initialize the random process $\mathcal{N}$ Receive initial state $s$ while not done do
        for each agent $i$, select action $a_i = \mu^{\theta_i}(o_i) + \mathcal{N}_t$ w.r.t the current policy and exploration;
        Execute action $a = (a_1, \dots, a_N)$, observe reward $r(s, a)$ and new state $s'$;
        Store transition $(s, a, r, s'$ in replay buffer $\mathcal{D}$;
        $s \leftarrow s'$;
        Initialize agent index $j = 1$;
        while $j < N$ do
            Sample a random mini-batch of $S$ samples $(s^j, a^j, r^j, s'^j) from \mathcal{D}$;
            Set $y^j = r_i^j + \gamma\, Q_i^{\phi'}(s'^j, a_1\prime, \dots, a'_N)\big|_{a'_k = \mu'_k(o_k^j)}$;
            Update critic by minimizing Bellman-error
              $\mathcal{L}(\phi_i) = \frac{1}{S}\sum_j \left(y^j - Q_i^{\phi}(s^j, a_1^j, \dots, a_N^j)\big|_{a_i = \mu_i(o_i^j)}\right)$;
            Update actor using the sampled policy gradient:
              $$\nabla_{\theta_i} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i}\mu_i(o_i^j)\nabla_{a_i}\, Q_i^{\phi_i}(s^j, a_1^j, \dots, a_N^j)\big|_{a_i = \mu_i(o_i^j)}$$
            $j \leftarrow j + 1$;
        end
        Update target network parameters for each agent $i$:
              $$\theta'_i \leftarrow \tau\theta_i + (1 - \tau)\theta'_i$$
        $i \leftarrow i + 1$;
    end
end
```

Algorithm 2: Multi-Agent Deep Deterministic Policy Gradient [Lowe et al., 2017]

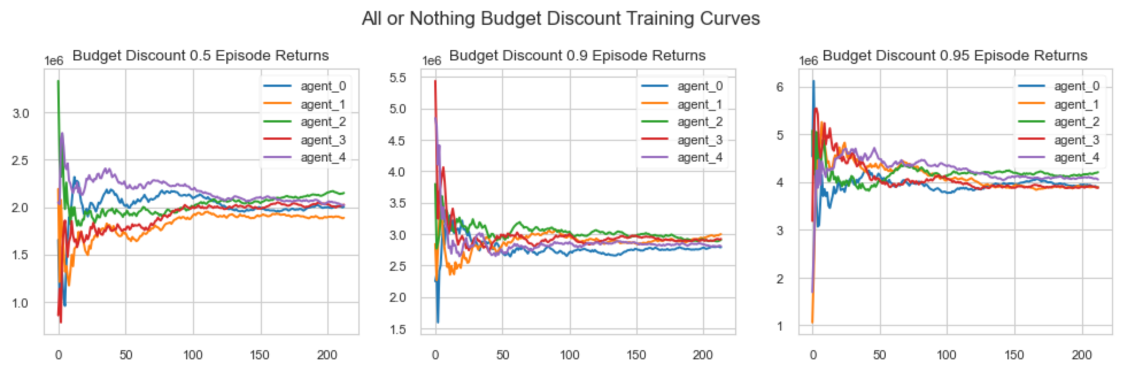

Figure 2: Episode total returns with a budget discount over time of 0.50 (left), 0.90 (middle), and 0.95 (right) when agents trade on all or nothing.

Furthermore, each agent seems to have converged to a similar critic function, just at different scales. This suggests the agents perceive the same actions lead to the same level of reward. In other words they have adopted the same strategy, and one that is overly pessimistic.

4.3 Nash Equilibrium and Determinism

It seems the deterministic nature of MADDPG and the centralized critic function led to a Nash Equilibrium that all the models converged to. Since the critic is centralized, the actor trains to find the best strategy while considering the strategy of all the other agents held constant. That is, the agent tries to improve its strategy until it can't anymore, while all other agents do the same. They all reach a strategy whereby they all try to buy in order to acquire more stocks

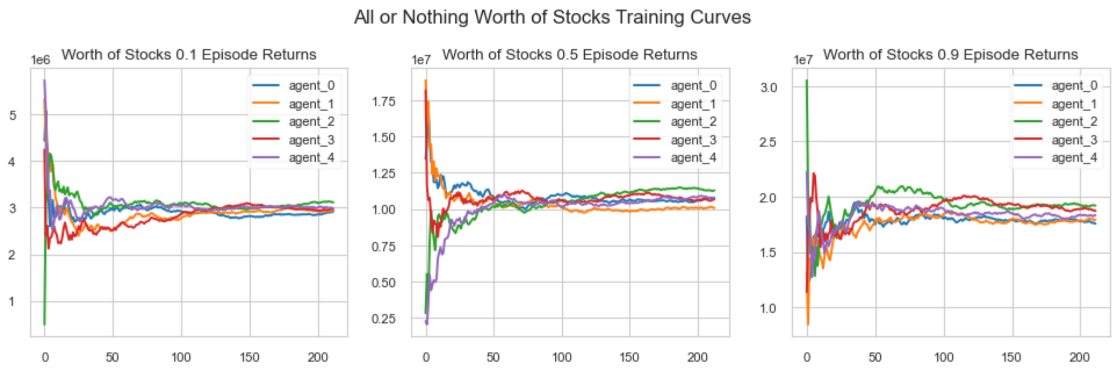

Figure 3: Episode total returns with a worth of stock of 0.10 (left), 0.50 (middle), and 0.90 (right) when agents trade on all or nothing.

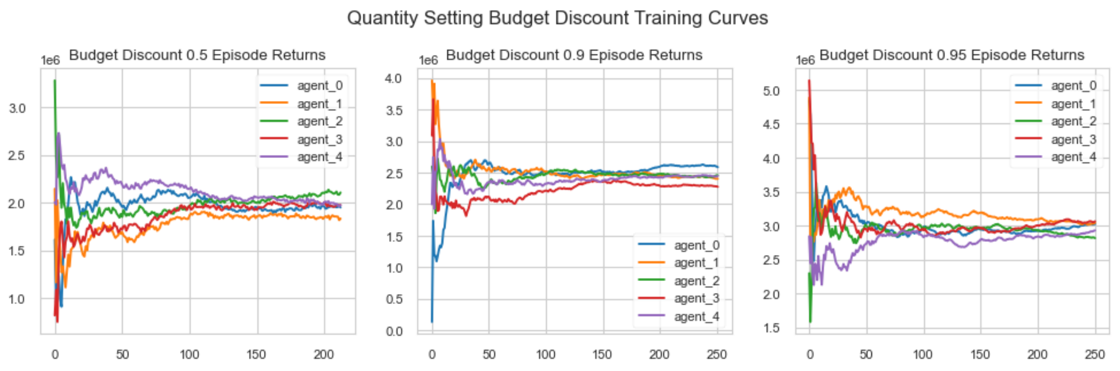

Figure 4: Episode total returns with a worth of stock of 0.10 (left), 0.50 (middle), and 0.90 (right) when agents trade with share quantities.

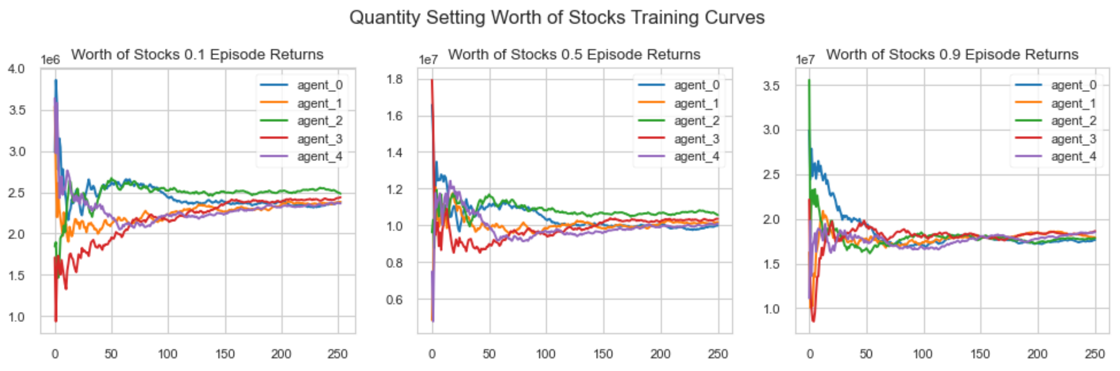

Figure 5: Episode total returns with a worth of stock of 0.10 (left), 0.50 (middle), and 0.90 (right) when agents trade with share quantities.

to later sell. But upon reaching this strategy, all the agents are stuck due to the deterministic policy. They all seek to buy in order to acquire more value, but the deterministic policy means they cannot randomly try another action but their best perceived one, buying. Furthermore, they cannot improve their strategy: if any agent chooses to sell instead of buy, then the remaining agents can thereafter sell or sit on the shares bought to accumulate value, and now the agent that has sold its shares is at a disadvantage. They have arrived at a Nash Equilibrium. Knowing the strategies of the other players, and treating the strategies of the other players as set in stone, no agent can benefit by changing their strategy.

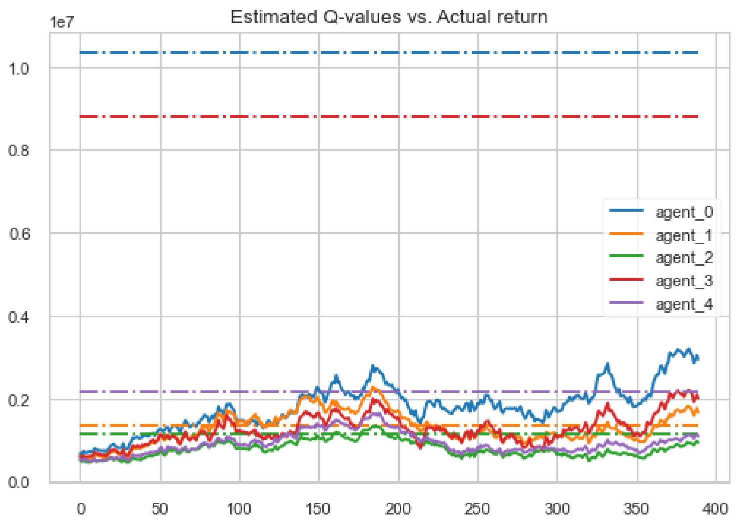

Figure 6: Q-values for one episode. Above is the quantity setting, worth of stocks = 0.90 model. All other models were similar. Dotted lines are agent's actual return

5 Conclusion

The agent based economic modelling approach is a promising one that may allow economists to bridge the gap in the aggregation problem. Many have, in this agent based approach, naturally tried to apply reinforcement learning in limited settings, such as abstracting away shares to be traded. However, two problems are still present in agent based approaches: computational complexity and determinism. Trying to limit the stochasticity of these models appears to in fact be impossible. The deterministic policies from reinforcement learning, while powerful, do not do well in such a complex environment, and can result in the agents getting stuck at a nash equilibrium.

References

Hyman P. Minsky. The financial instability hypothesis. *Levy Economics Institute of Bard College, Annandale-on-Hudson, NY*, 1992.

Douglas W. Blackburn and Andrey Ukhov. Individual vs. aggregate preferences: The case of a small fish in a big pond. *SSRN*, 2008. doi:http://dx.doi.org/10.2139/ssrn.941126.

Jordi Galí. *Monetary policy, inflation, and the business cycle: an introduction to the new Keynesian framework and its applications*. Princeton University Press, 2015.

Muaz Niazi and Amir Hussain. Agent-based computing from multi-agent systems to agent-based models: a visual survey. *Scientometrics*, 89(2):479–499, 2011.

Johann Lussange, Ivan Lazarevich, Sacha Bourgeois-Gironde, Stefano Palminteri, and Boris Gutkin. Modelling stock markets by multi-agent reinforcement learning. *Computational Economics*, 57(1):113–147, 2021.

Seungmoon Song, Łukasz Kidziński, Xue Bin Peng, Carmichael Ong, Jennifer Hicks, Sergey Levine, Christopher G Atkeson, and Scott L Delp. Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation. *Journal of neuroengineering and rehabilitation*, 18(1):1–17, 2021.

Jae Won Lee. Stock price prediction using reinforcement learning. In *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No. 01TH8570)*, volume 1, pages 690–695. IEEE, 2001.

Sutta Sornmayura. Robust forex trading with deep q network (dqn). *ABAC Journal*, 39(1), 2019.

Ömer Faruk Ertuğrul and Mehmet Emin Tağluk. Forecasting financial indicators by generalized behavioral learning method. *Soft Computing*, 22(24):8259–8272, 2018.

Marco Corazza and Andrea Sangalli. Q-learning and sarsa: a comparison between two intelligent stochastic control approaches for financial trading. *University Ca'Foscari of Venice, Dept. of Economics Research Paper Series No*, 15, 2015.

João Carapuço, Rui Neves, and Nuno Horta. Reinforcement learning applied to forex trading. *Applied Soft Computing*, 73:783–794, 2018.

Leonardo Conegundes and Adriano C Machado Pereira. Beating the stock market with a deep reinforcement learning day trading system. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

Callum Tilbury. Reinforcement learning in macroeconomic policy design: A new frontier? *ArXiv*, 2022. doi:https://doi.org/10.48550/arXiv.2206.08781.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *ArXiv*, 2017. doi:https://doi.org/10.48550/arXiv.1706.02275.