Investigating, Predicting, and Mitigating Collusive Behavior in Deep Reinforcement Learning-Based Pricing AIs

Inauguraldissertation
zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften
der Universität Mannheim

vorgelegt von

Michael Schlechtinger

Mannheim, 2024

Dekan:	Prof. Dr. Claus Hertling, Universität Mannheim
Referent:	Prof. Dr. Heiko Paulheim, Universität Mannheim
Korreferent:	Prof. Dr. Heiner Stuckenschmidt, Universität Mannheim

Abstract

In the rapidly evolving landscape of eCommerce, dynamic pricing using Artificial Intelligence (AI) has become increasingly prevalent. Pricing AIs, particularly those utilizing (deep) Reinforcement Learning, continuously adjust to dynamic market conditions, raising concerns about potential market collusion. This thesis addresses this issue through several approaches.

Firstly, we investigate a modified prisoner's dilemma scenario where three agents play rock-paper-scissors. Results indicate potentially collaborative behavior characterized by an action selection dissectable into specific stages, establishing the possibility of developing collusion prevention systems that can recognize situations that might lead to collusion between competitors. We provide evidence that agents can perform tacit cooperation strategies without being explicitly trained to do so.

Second, this research employs an experimental oligopoly model of repeated price competition, systematically varying the environment to cover scenarios from basic economic theory to subjective consumer demand preferences. We explore strategies and emerging pricing patterns leading to collusion, including scenarios where agents cannot observe competitors' prices. Comprehensive legal analysis is provided across all scenarios.

Third, we examine the agents' ability to use pricing information to predict their competitors' behavior. Thus, predictive statistical techniques and time series analysis methodologies are employed to anticipate collusive outcomes. Findings indicate that self-learning pricing algorithms' convergence towards a collusive market outcome can be accurately anticipated using machine learning methodologies.

Finally, we develop a method to mitigate predictive and supracompetitive pricing using a combination of various training strategies. By implementing a supervision algorithm penalizing collusion and incentivizing competitiveness we effectively incentivize the agents to act competitively rather than collaboratively. The results demonstrate that the convergence of self-learning pricing algorithms towards collusive outcomes can be accurately predicted and mitigated in real time.

Zusammenfassung

Dynamische Preisgestaltung mit Hilfe künstlicher Intelligenz verändert die E-Commerce Landschaft in einer rasanten Geschwindigkeit. Preisalgorithmen, auf Basis von (Deep) Reinforcement Learning, passen sich kontinuierlich an dynamische Marktbedingungen an, was Bedenken hinsichtlich potenzieller Marktabsprachen weckt.

Zur Untersuchung dessen wird zu Beginn dieser Dissertation ein Szenario entwickelt, in dem drei Agenten Stein-Papier-Schere spielen. Die Ergebnisse zeigen, dass die Aktionsauswahl der Agenten in spezifische Phasen aufgeteilt werden kann. Darüber hinaus zeigen die Ergebnisse, dass Agenten in der Lage sind, stillschweigende Kooperationsstrategien zu etablieren, ohne dies explizit gelernt zu haben.

Daraufhin wird ein experimentelles Oligopolmodell erarbeitet, welches systematisch variiert wird, um Szenarien der grundlegenden Wirtschaftstheorie bis hin zu subjektiven Nachfragepräferenzen der Verbraucher abzudecken. Zusätzlich untersucht diese Studie Szenarien, in welchen Marktteilnehmer die Preise ihrer Konkurrenz nicht einsehen können. Auch hier konnten trotz der Einschränkung Absprachestrategien entwickelt und kollusive Preismuster aufgezeigt werden. Die Datenwissenschaftliche Analyse wird mithilfe juristischer Einblicke ergänzt.

Drittens wird untersucht, inwiefern die Agenten Preisinformationen nutzen können, um das Verhalten ihrer Konkurrenten vorherzusagen. Dazu werden Methoden der prädiktiven Statistik, sowie der Analyse von Zeitreihen verwendet. Die Ergebnisse zeigen, dass die Konvergenz selbstlernender Preisalgorithmen in Richtung eines kollusiven Marktergebnisses präzise vorhergesagt werden kann.

Schließlich wird eine Methode entwickelt, um kollusive und suprakompetitive Preisbildung durch eine Kombination verschiedener Trainingsstrategien zu lindern. Dieser Ansatz schafft einen wettbewerbsfähigen Markt und implementiert einen Kontrollalgorithmus, der gesetzeswidrige Absprachen bestraft und Anreize für Wettbewerb schafft. Die Ergebnisse zeigen, dass die Konvergenz selbstlernender Preisbildungsalgorithmen zu kollusiven Ergebnissen mithilfe gewisser Methodiken vorhergesagt und somit verhindert werden kann.

Acknowledgements

There have been several people who majorly influenced my Ph.D. journey. I would like to take this opportunity to thank the ones who have given me a formidable amount of support throughout the last 4 years.

First and foremost, I would like to express my deepest gratitude to Heiko Paulheim. Despite my research topic being quite unique within a group primarily focused on knowledge graphs and natural language processing, Heiko's remarkable ability to challenge my way of thinking was invaluable. Although I had to navigate fields such as Reinforcement Learning, Economics, and Law independently, Heiko was always available whenever I encountered obstacles in my research. With a few questions, he was generally able to help me gain a broader perspective by encouraging me to think outside the box. This guidance often led to great solutions (although sometimes it sparked even more questions).

This dissertation would not have been possible without the knowledge and unwavering support of my dear friend and co-author, Damaris Kosack. We began as strangers working on the KarekoKI project, but gradually grew into a strong team. Damaris imparted invaluable insights into the legal aspects of our research, and, more importantly, taught me the precision and power of language. Thinking like a law scholar made me appreciate the skill of taking a step back and interpreting sequences of words from a different perspective. Having Damaris by my side for our numerous conference presentations alleviated many concerns, as she seemed to be able to confidently reply to respond to any feedback. Beyond her remarkable legal expertise, she has always been there to offer general advice. In hindsight, I feel like we have formed a formidable team, and I wish her all the best in her future endeavors, wherever her curiosity may lead.

My Ph.D. journey in Mannheim began right after the first Covid lockdown in Germany. This situation, particularly in the beginning, made it challenging to have in-depth face-to-face conversations with my colleagues. Despite these challenges, my colleagues made me feel welcome and kept me sane from the very first day at the university. Adrian, Andreea, Franz, Nico, and Sven (listed alphabetically to avoid any ranking) always endeavored to help me find solutions to the research questions I was grappling with, even though their research fields were only tangentially related to mine. Not limited to, but especially throughout many early lunches, I got to appreciate the presence of these like-minded individuals.

Finally, I would like to express my love and gratitude to my partner Anouk, who not only endured my monologues disguised as dialogues to help me articulate my thoughts but also supported me through every decision in my life. I also want to thank my parents, my sister, and my friends, who have always been there for me, no matter if I felt high or low. Thank you so much.

- Michael

Contents

List of Figures

List of Tables

List of Acronyms

ACF Autocorrelation Function

AI Artificial Intelligence

ARC Act against Restraints of Competition

AR Autoregressive

CB Competitive Benchmark

CMDP Continuous Markov Decision Process

DDPG Deep Deterministic Policy Gradients

DQN Deep Q-Network

DRL Deep Reinforcement Learning

ECJ European Court of Justice

E-Commerce Electronic Commerce

EU European Union

FFT Fast Fourier Transform

HER Hindsight Experience Replay

LOWESS Locally Weighted Scatterplot Smoothing

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

MARL Multi-Agent Reinforcement Learning

MAS Multi-Agent Systems

MDP Markov Decision Process

ML Machine Learning

MMDP Multi-Agent Markov Decision Process

MP Monopolistic Price

MSE Mean Squared Error

PACF Partial Autocorrelation Function

PAAMS Practical Applications of Agents and Multi-Agent Systems

POMDP Partially Observable Markov Decision Process

PPO Proximal Policy Optimization

Q-Learning Quality Learning

RPS Rock Paper Scissors

RL Reinforcement Learning

RMSE Root Mean Squared Error

SARSA State-Action-Reward-State-Action

TD Temporal Difference

TFEU Treaty on the Functioning of the European Union

TRPO Trust Region Policy Optimization

WCLR Win-Continue, Lose-Reverse

Part I

Motivation and Fundamentals

Chapter 1

Introduction

Artificial Intelligence (AI) has seamlessly integrated into various facets of our daily lives, from personal assistants to autonomous vehicles. A significant yet less visible application of AI is in the realm of E-commerce, where companies leverage AI to optimize pricing strategies and maximize profits. The transition from static heuristics to dynamic, AI-driven pricing algorithms has revolutionized the industry, yielding substantial improvements in average daily profits [83, 107].

Pricing algorithms automatically adjust prices based on numerous factors such as demand, competition, and inventory levels. These algorithms, increasingly powered by Deep Reinforcement Learning (DRL), learn optimal pricing strategies through trial and error, adapting to market conditions in real-time. Studies have already shown that in certain markets it is unreasonable to conduct business at a profitable level lacking this technology, as algorithmic price setting frequencies create a significant competitive edge [12].

However, the rapid adoption of AI-driven pricing algorithms has raised concerns about their impact on market competition and consumer welfare. One significant issue is the potential for these algorithms to engage in tacit collusion, leading to higher prices and reduced competition. Unlike traditional collusion, which involves explicit agreements between competitors, algorithmic collusion can occur without direct human intervention, making it challenging to detect and regulate. From a doctrinal perspective, the challenge lies in the difficulty of attributing legal responsibility directly to a market outcome [31].

The emergence of collusive market outcomes through AI-based pricing algorithms poses substantial legal and ethical questions. If such outcomes result from coordinated actions, they could violate antitrust laws, specifically Article 101 of the Treaty on the Functioning of the European Union (TFEU). This thesis aims to explore the factors contributing to the collusive tendencies of DRL agents and assess the associated risks and vulnerabilities in their deployment. By examining the underlying mechanisms, we seek to enhance the transparency and comprehensibility of AI-driven pricing strategies.

In summary, while DRL-based pricing algorithms offer significant benefits in profitability and market efficiency, their potential to harm competition and consumer welfare cannot be ignored. This dissertation sheds light on these critical issues, contributing to a more comprehensive understanding of the legal and economic implications of AI-driven pricing strategies.

1.1 Research Questions

The main goal of this thesis is to investigate the factors that lead to algorithmic collusion and to develop methods for its prevention. This section outlines the key research questions that guide this investigation, aiming to uncover the underlying mechanisms of algorithmic collusion and explore the legal and regulatory frameworks needed to address this emerging challenge.

- **RQ1** *How can DRL agents learn to collaborate and establish collusive states in competitive scenarios, and what tools can be used to measure and predict such behaviors?* This consolidated research question explores the mechanisms through which DRL agents might achieve cooperation in competitive environments, the established strategies necessary to result in collusive states, and the development and application of analytical tools to quantify and predict algorithmic collusion. The detailed analysis of these aspects is addressed throughout Parts I and II of this thesis.

- **RQ2** *Can DRL agents learn to collude without explicitly communicating?* The ability of DRL agents to collude implicitly, without direct communication, is a critical area of study. This question examines the subtle interactions and patterns that might lead to such behavior, with insights provided in Chapter 4.

- **RQ3** *How can we mitigate algorithmic collusion and supracompetitive pricing?* If DRL agents gain the ability to establish equilibria within collusive states, we need to investigate methods to mitigate and possibly prevent these circumstances. This question is addressed through various approaches detailed in Part II, with comprehensive strategies highlighted in Chapter 6.

1.2 Contributions

This thesis presents a variety of contributions across several areas of Multi-Agent DRL, pricing algorithms, collusion detection, and prevention. Specifically, the key contributions are:

- **We highlight emerging pricing patterns that agents seem to follow to achieve a collusive outcome.** Understanding these patterns is crucial for developing strategies to prevent algorithmic collusion. This contribution is thoroughly explored in Chapters 4 and 5.
- **We investigate a scenario where agents achieve a collusive outcome without the ability to observe their competitors' prices.** This scenario examines how limited information affects agent behavior and collusion, providing valuable insights in Chapter 4.
- **We provide a legal perspective and analysis specific to the establishment of collusive outcomes in DRL pricing agents' based economies.** This analysis addresses the legal implications and regulatory challenges associated with DRL-induced collusion, detailed in Chapter 5.
- **We propose a novel demand framework that enables the implementation of various demand models, allowing for a weighted blending of different models.** This framework offers flexibility and accuracy in modeling demand, which is essential for dynamic pricing strategies. This environment furthermore allows for extensive testing and analysis of DRL agents in various market scenarios, enhancing the robustness and applicability of the research findings. The foundation of the framework is established in Chapter 4 and subsequently utilized throughout Parts II and III.
- **We extend the pricing agents' research analysis by examining deep learning algorithms, i.e., PPO and DQN.** Most of the current research within this realm primarily focused on basic Q-learning. By incorporating advanced deep learning algorithms, this research extends the current understanding and application of DRL in pricing strategies, as detailed throughout Parts I and II.
- **We examine the impact of varying market entry timings on collusion.** By analyzing how the timing of market entry affects collusive behavior, we provide deeper insights into the dynamics of DRL-based pricing strategies, as detailed in Appendix Chapter B.

- **We analyze the influence of different neural network structures on DRL performance.** This contribution explores how variations in neural network architectures, specifically the number of neurons and layers of the NN, can impact the effectiveness and efficiency of DRL agents, with findings presented in Appendix Chapter B.
- **We create a factor to measure collusiveness and predict the upcoming behavior of DRL-based pricing agents.** This contribution involves developing methods to measure and forecast collusive behavior among DRL agents, with extensive methodologies and results presented in Chapter 5.
- **We prevent collusion using sparse rewards.** Drawing from safe exploration DRL literature, we investigate and test multiple approaches to prevent collusive outcomes. Furthermore, we provide a solution to create new optima and thus prevent agents from quickly settling on a global, collusive optimum. The prevention strategies are outlined in Chapter 6.

In addition to the contributions mentioned above, the codebase of the simulation environment, along with all its extensions, is provided in the form of a GitHub repository[1].

1.3 Thesis Outline

In the following, we specify the structure of the thesis.

Part I: Motivation and Fundamentals
This part introduces the research context, underlying theories, and essential concepts required for understanding the subsequent chapters. It lays the groundwork by discussing Reinforcement Learning (RL), dynamic pricing, antitrust law, and economics.

Chapter 1: Introduction
This chapter introduces the context and significance of the research. It outlines the key research questions and contributions of the thesis, providing an overview of the structure and objectives of the dissertation.

Chapter 2: Fundamentals
This chapter provides the necessary background information on RL, dynamic pricing, antitrust law, and economics. It covers the basics of RL,

[1]`https://github.com/mschlechtinger/PriceOfAlgorithmicPricing/`

including key algorithms, and delves into dynamic pricing strategies in e-commerce, relevant legal frameworks, and economic principles.

Chapter 3: Investigating Collusion in a Toy Problem
This chapter explores collusion in a simplified environment. It discusses related work on trustworthy AI, the cartel prohibition, and multi-agent reinforcement learning. The methodology and implementation of the toy problem are presented, followed by results and discussions on collusive behavior.

Part II: Investigate, Predict, and Prevent Collusion in a Market Simulation
This part extends the insights from Part I to a more complex market simulation. It focuses on investigating, predicting, and preventing collusive behavior among DRL agents in a competitive market environment.

Chapter 4: Collusion in a Market Simulation
This chapter examines collusive behavior in a market simulation. It introduces the experimental framework, which is vital to each of the following analyses. It thus establishes the Markov Decision Process and analyzes the pricing algorithms' behavior in multiple scenarios. Results from different scenarios are analyzed to understand how agents might achieve collusion.

Chapter 5: Predicting and Quantifying Collusion
This chapter focuses on developing models to predict and quantify collusion among DRL agents. It presents classification, regression, and time series analysis techniques to measure collusive behavior and evaluates their effectiveness through extensive experiments.

Chapter 6: Preventing Collusion
This chapter uses the analysis toolkit provided in Chapter 5 to address methods to prevent collusion among DRL agents. It thus proposes approaches such as price manipulation and supervision reward factors, presenting experimental results and discussions on the effectiveness of these methods in mitigating collusive outcomes.

Part III: Conclusion and Outlook
This part summarizes the findings of the research, discusses the implications, and provides directions for future work.

Chapter 7: Conclusion and Outlook
This chapter summarizes the key findings of the research and discusses the implications of the results. It also provides an outlook on future research

directions, including potential advancements in empirical data analysis, investigation of confounding factors, and the development of transparent and explainable DRL models.

Appendix
The appendix includes additional experimental results, ablation studies, and detailed settings for the algorithms used in the research.

1.3.1 Publications

The following peer-reviewed papers contribute to this thesis:

- **Winning at Any Cost - Infringing the Cartel Prohibition With Reinforcement Learning**

  Schlechtinger, Michael, Damaris Kosack, Heiko Paulheim, and Thomas Fetzer. In Advances in Practical Applications of Agents, Multi-Agent Systems, and Social Good. The PAAMS Collection. Salamanca, Spain: Springer Cham, 2021. [114]

- **How Algorithms Work and Play Together**

  Schlechtinger, Michael, Damaris Kosack, Heiko Paulheim, and Thomas Fetzer. Concurrences: Revue Des Droits de La Concurrence 2021, no. 3, Article 102088 (2021): 19–23. [113]

- **The Price of Algorithmic Pricing: Investigating Collusion in a Market Simulation with AI Agents**

  Schlechtinger, Michael, Damaris Kosack, Heiko Paulheim, Thomas Fetzer, and Franz Krause. In Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, 2748–50. AAMAS '23. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2023. [115]

- **By Fair Means or Foul: Quantifying Collusion in a Market Simulation with Deep Reinforcement Learning**

  Schlechtinger, Michael, Damaris Kosack, Franz Krause, and Heiko Paulheim. arXiv, 4 June 2024. Forthcoming in the Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Jeju, South Korea, August 3–9, 2024. [112]

Chapter 2

Fundamentals

2.1 Reinforcement Learning

Reinforcement Learning is the study of agents and how they learn by trial and error [3].

The fact that we learn through interacting with our environment is a fundamental concept in understanding knowledge acquisition and intelligence [81]. This behavior can commonly be observed in infants, especially evident when they start to crawl. The exploration process serves a crucial role in their understanding of objects and spatial awareness. As they move and interact with their environment, they gather valuable insights about both objects and the space they inhabit [25][1].

RL emulates this learning behavior by defining the following key components (cf. Figure 2.1):

- **Agent** and **Environment**: The agent is the decision-maker, and the environment includes everything the agent interacts with.
- **Action spaces**: Choices the agent can make.
- **States** and **Observations**: The current situation of the agent within the environment.
- **Rewards**: Feedback from the environment following an action. The goal is to maximize cumulative rewards.
- **Policies**: Strategies for choosing actions.

[1]For an in-depth introduction into RL, please refer to [129]

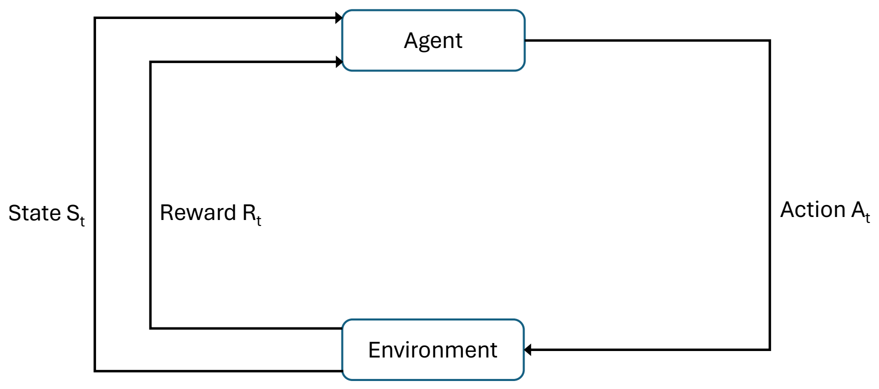


Figure 2.1: Key concepts in Reinforcement Learning.

- **value functions**: Estimations of expected rewards.
- **Trajectories**: Sequences of states and actions.

However, instead of speculating on how learning occurs in humans or animals, RL focuses on simulated learning scenarios to test different learning methods. The methodology deviates from traditional machine learning approaches with a focus on goal-directed learning from interaction. RL evaluates various learning methods within idealized scenarios, emphasizing designs for machines that solve scientific or economic learning problems through mathematical analysis or computational experiments [129].

2.1.1 Markov Decision Process

Building on the foundational understanding of RL, where agents learn through interaction with their environments to achieve goal-directed behavior, we delve deeper into the mathematical underpinnings that facilitate this learning process. Markov Decision Processes (MDPs) serve as a critical framework in this context, offering a formalized approach to modeling the decision-making environment of agents. MDPs encapsulate the dynamics of RL's key components within a structured mathematical model that emphasizes the probabilistic nature of transitioning between states and the optimization of cumulative rewards over time [129, 141].

An MDP is defined by:

- A set of states S,
- A set of actions A,

- A transition function $T(s, a, s')$ which defines the probability of transitioning from state s to state s' after taking action a,
- A reward function $R(s, a, s')$ which defines the immediate reward received after transitioning from state s to state s' by taking action a,
- A discount factor γ which represents the importance of future rewards.

In the context of pricing agents, states can represent different snapshots of the market prices, actions can be different pricing strategies, and rewards can reflect the profit or market share captured as a result of these strategies. The transition function models the dynamics of the market, including how competitors' actions (and external factors) influence the market state. Properly defining this MDP is pivotal to exploring strategies that agents might employ to maximize their cumulative rewards, which in turn may lead to coordinated pricing strategies that resemble collusion or cartel behavior.

Extensions and Variations of MDPs

In many practical applications, the state and action spaces are continuous rather than discrete. Continuous MDPs (CMDPs) extend the traditional MDP framework by allowing state and action spaces to be represented as continuous variables. This extension is crucial for applications such as robotic control and financial modeling, where the granularity of state and action spaces can significantly impact performance [16].

Multi-agent systems (MAS) involve multiple decision-making agents interacting within the same environment. This extension introduces additional complexity, as the state transition dynamics now depend on the actions taken by all agents in the system. These interactions can be modeled using Stochastic Games or Multi-Agent MDPs (MMDPs) [123]. Understanding how agents coordinate or compete within these frameworks is essential for applications such as market simulations and autonomous vehicle coordination.

While MDPs provide a robust framework for decision-making under uncertainty, they assume that the agent has complete knowledge of the current state. In a market scenario, this assumption does not necessarily hold, as agents often have limited or noisy observations of the true state. A partially Observable Markov Decision Process (POMDP) addresses these limitations by introducing a new set of components:

- A set of observations O, which the agent can perceive from the environment,
- An observation function $O(o|s, a)$ which defines the probability of observing o given that the agent took action a and the environment is in state s.

2.1.2 Multi-Agent Reinforcement Learning

Multi-Agent Reinforcement Learning (MARL) extends the fundamental components of RL agents, environments, states, actions, rewards, policies, and value functions to accommodate multiple agents. Each agent aims to maximize its own cumulative reward while adapting to the behaviors of other agents. The interactions among agents introduce additional layers of complexity and strategic depth, as each agent's optimal strategy may depend on the strategies of others. Key concepts in MARL include:

- **Joint Action Spaces**: The combined set of all possible actions that can be taken by all agents in the environment.
- **Joint States**: The combined states resulting from the interactions of all agents with the environment.
- **Reward Sharing**: Mechanisms for distributing rewards among agents based on their individual and collective actions.
- **Coordination and Competition**: The dual aspects of MARL where agents may need to cooperate to achieve common goals or compete to maximize individual rewards.

MARL presents several unique challenges compared to single-agent RL. The learning environment becomes non-stationary from the perspective of any single agent due to the evolving policies of other agents, complicating the learning process. Additionally, the state and action spaces can grow exponentially with the number of agents, making scalability a significant issue. Ensuring effective communication and coordination among agents is challenging, especially in cooperative tasks where they need to work together to achieve common goals. Furthermore, balancing the exploration of new strategies with the exploitation of known effective strategies becomes more complex, as the actions of one agent can influence the rewards and outcomes of others, creating a highly interdependent environment.

MARL is particularly relevant in economic simulations where multiple entities, such as firms, interact within a market environment. Inside of this environment, we can model agents as firms that learn to set prices and adjust their strategies based on competitors' actions, potentially leading to collusive outcomes. MARL in this context allows for an exploration of the strategic interactions between the agents.

2.1.3 Bellman Equation

The Bellman equation is essential in RL for formulating the dynamic programming approach that solves decision-making problems by breaking them down into smaller, manageable sub-problems, thus encapsulating the principle of optimality for decision-making under uncertainty. It plays a critical role in determining the optimal policy π by relating the value of a state $v(s_t)$ to the values of subsequent states $(v(s_{t+1}, v(s_{t+2}), ...)$, guiding agents towards decisions that maximize cumulative rewards R [13]. The formulation of the equation is adaptable to the specific nature of the environment, whether the setting is predictable (deterministic) or involves randomness (stochastic), to fit the problem at hand.

Deterministic Bellman Equation

In a deterministic setting, the value of a state s denoted $V(s)$, simplifies to the following equation:

$$V(s) = \max_a \big(R(s,a) + \gamma V(s')\big) \tag{2.1}$$

where:

- $R(s,a)$ is the reward the agents receive after taking action a in state s.
- γ is the discount factor, which values future rewards less than immediate rewards.
- $V(s')$ is the value of the state resulting from taking action a in state s.

This equation assumes that for every state s and action a, the transition to the next state s' is deterministic, and the reward $R(s,a)$ is also deterministic.

Stochastic Bellman Equation

Consider a state s in a finite MDP. The value of s, denoted $V(s)$, can be expressed as follows:

$$V(s) = \max_a \left(R(s,a) + \gamma \sum_{s'} P(s,a,s')V(s')\right) \tag{2.2}$$

where:

- $P(s,a,s')$ is the probability of transitioning to state s' from state s after taking action a.

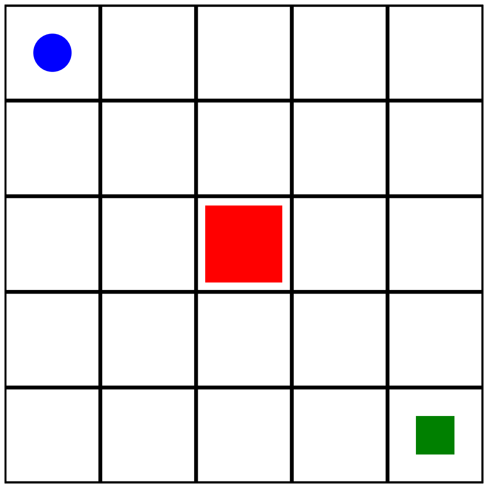


Figure 2.2: Grid World example containing the agent (blue), a punishment tile (red), and the goal (green).

- $R(s, a)$ is the immediate reward received after transitioning from state s to state s', due to action a.
- γ is the discount factor, which values future rewards less than immediate rewards.
- The sum over $s' \in \mathcal{S}$ represents summation over all possible subsequent states.

2.1.4 Grid World

A classic example of an RL problem is *Grid World*, a finite, two-dimensional, discrete environment [129] (cf Figure 2.2). In Grid World an agent (blue) navigates through various states—represented as grid cells—to reach a goal, thereby encountering rewards (green) and obstacles (red). This simple yet effective model allows for a comprehensive discussion of the previously mentioned RL terminology. In this example, The agent, as the decision-making entity, explores the environment (the grid), takes actions (moves north, east, south, or west), and receives rewards (+1 for reaching the goal, -1 for attempting to move outside the grid, -10 for stepping on a red tile representing a hazard). While navigating through the grid, the agent constantly reevaluates which action will guide it to a state closer to the goal, essentially leading toward optimal decisions. This function quantifies the desirability of state-action pairs. Based on this information, the agent develops a policy, which essentially translates the value function into actionable decisions. The policy, developed by the agent, is a direct mapping from states to actions, dictating the agent's movement strategy. An optimal policy maximizes the expected cumulative reward, which is often the sum of the rewards the agent collects, discounted by a factor that accounts for the uncertainty of future rewards. The agent then uses a trial-and-error learning process, often employing specific algorithms such as Q-learning or policy gradients, to discover which actions yield the highest reward by updating the policy based on the value function's estimations.

Moreover, the policy can be deterministic or stochastic. In a deterministic Grid World example, like the one in Figure 2.2, the outcomes of all actions are known and predictable. In a stochastic environment certain states have varying rewards and transition dynamics are subject to change. As the agent interacts with the environment, the policy becomes more refined, increasingly leading the agent towards the goal state while avoiding penalties and maximizing the overall reward.

If we assume a deterministic environment of Grid World, a respective MDP can be defined as a tuple $M = (S, A, T, R, \gamma)$, where:

- S is a finite set of states,
- A is a finite set of actions,
- $T : S \times A \times S \to [0, 1]$ is the state transition probability function,
- $R : S \times A \times S \to \mathbb{R}$ is the reward function, and
- γ is the discount factor, $\gamma \in [0, 1]$.

Given a state $s_t \in S$, an action $a_t \in A$, and a resulting state $s_{t+1} \in S$, the transition probability $T(s_t, a_t, s_{t+1})$ specifies the likelihood of transitioning from s_t to s_{t+1} by taking action a_t.

$$T(s_t, a_t, s_{t+1}) = \begin{cases} 1 & \text{if } a_t \text{ leads from } s_t \text{ to } s_{t+1} \text{ directly without obstacles,} \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

The reward function $R(s_t, a_t, s_{t+1})$ specifies the immediate reward received after transitioning from s_t to s_{t+1} via action a_t. The discount factor γ quantifies the importance of future rewards.

$$R(s_t, a_t, s_{t+1}) = \begin{cases} +1 & \text{if } s_{t+1} \text{ is the goal state,} \\ -1 & \text{if } a_t \text{ leads outside the grid,} \\ -10 & \text{if } s_{t+1} \text{ is a hazard state (red tile),} \\ -0.1 & \text{otherwise.} \end{cases} \tag{2.4}$$

2.1.5 Reinforcement Learning Algorithms

The algorithms operationalizing these RL principles serve as the core mechanisms by which an agent can learn from and interact with its environment. While the RL field encompasses a diverse array of algorithms, certain foundational techniques are critical to understanding the standard RL problem-solving toolkit. Among these, *Q-Learning* and *policy gradient* methods stand out due to their broad applicability [144, 131]. This section delves into five fundamental RL algorithms, detailing their mechanisms, advantages, and typical applications.

RL approaches are categorized into two primary types: those that operate without an explicit model of the environment (model-free) and those that leverage a constructed or assumed model to make decisions (model-based) [73]. Model-free algorithms, such as Q-learning and SARSA, operate without an explicit model of the environment. They learn the optimal policy through trial and error, directly from the rewards and transitions experienced. This makes them well-suited for environments where the dynamics are unknown or difficult to model. On the other hand, model-based algorithms attempt to construct a model of the environment's dynamics—understanding how actions lead to subsequent states and rewards. This model is then used to plan and make decisions. Model-based approaches can be more sample efficient, as they allow for simulated experiences to supplement actual interactions, but they require accurate models to be effective, which can be challenging to obtain in complex environments. In the remainder of this dissertation, our selection is confined to model-free methods as these are generally more popular, quicker to implement, and more extensively developed and tested than model-based methods [3], which lend themselves for experimental setups.

Monte Carlo Methods

Monte Carlo Methods are one of the first learning methods for solving decision-making problems under uncertainty. After collecting enough experiences (i.e., by the completion of an episode), this method samples and averages the returns for each state-action pair. Unlike dynamic programming approaches, Monte Carlo methods do not require a model of the environment and can learn directly from episodes of experience [130].

The value of a state s, denoted $V(s)$, is estimated as the average of returns G_t following visits to s, where the return G_t is the total discounted reward from time t:

$$V(s) = \frac{1}{N(s)} \sum_{t \in \mathcal{T}(s)} G_t, \tag{2.5}$$

where $N(s)$ is the number of times state s has been visited, and $\mathcal{T}(s)$ is the set of time steps at which s was visited. For incremental updates, the formula incorporates α as follows:

$$V(s) \leftarrow V(s) + \alpha(G_t - V(s)), \tag{2.6}$$

where $G_t - V(s)$ is the error in the current estimate. The learning rate, α, is a crucial hyperparameter in Monte Carlo methods and other RL algorithms in general. It determines the weight given to new information relative to old information. A higher α allows the algorithm to learn more quickly, adapting to new data at the expense of potentially disregarding valuable older information. Conversely, a lower α makes learning from new data more gradual, preserving older information but possibly slowing down the adaptation to new patterns.

In this context of the simple grid world environment, Monte Carlo methods can be employed to estimate the value of each state, representing the expected return or cumulative reward of starting from that state and following a certain policy to reach the goal. Each episode represents a complete journey from the start state to the goal or until a predetermined number of steps is reached. After collecting a sequence of states, actions, and rewards for each episode, the agent calculates the return for each state visited during the episode. This process is repeated for multiple episodes, allowing the agent to improve its policy for navigating the grid world.

A primary advantage of Monte Carlo Methods is their straightforward implementation and the ability to learn directly from episodes of experience without requiring a model of the environment. This makes Monte Carlo methods particularly appealing for problems where the environment's dynamics are complex or unknown. Moreover, by focusing on complete episodes, these methods can effectively handle episodic tasks where the goal is to optimize the return over a finite sequence of steps. However, their reliance on episodes to complete before updating value estimates can lead to slower learning, especially in environments where episodes are long or infrequent. Additionally, because Monte Carlo methods average returns over entire episodes, they can exhibit high variance in their estimates, requiring many episodes to converge to stable value function estimates. Balancing the exploration of new strategies with the exploitation of known strategies remains a critical challenge, as excessive exploration can lead to suboptimal learning outcomes [130]. These shortcomings set the stage for methods that learn from incomplete sequences.

Temporal Difference (TD)

Temporal Difference (TD) Learning blends ideas from Monte Carlo methods and Dynamic Programming. It stands out for its ability to learn directly from raw experience without a model of the environment's dynamics, and yet, it does not require the episode to complete before updating value estimates [128]. Unlike Monte Carlo methods, which wait until the end of an episode to determine the total reward and update value function estimates, TD Learning uses bootstrapping[2] to update estimates based in part on other learned estimates, without waiting for a final outcome. This approach allows for more frequent updates and can lead to faster learning in many scenarios.

Central to TD Learning is the TD error, a measure of the difference between consecutive value function estimates. For a given state s_t and its successor state s_{t+1}, the TD error δ_t is given by:

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t), \tag{2.7}$$

where r_{t+1} is the reward received after transitioning from state s_t to state s_{t+1}, γ is the discount factor, and $V(s_t)$ and $V(s_{t+1})$ are the value estimates of the current and next states, respectively. The value function for state s_t is then updated as follows:

$$V(s_t) \leftarrow V(s_t) + \alpha\delta_t, \tag{2.8}$$

where α is the learning rate, a parameter controlling the magnitude of value function updates.

Going back to the grid world example, where an agent must navigate from a starting position to a goal. Using TD Learning, the agent updates its value function after each step based on the TD error, gradually learning the expected total reward from each position in the grid. This process enables the agent to develop a policy that guides it along the shortest path to the goal. TD learning's introduction to bootstrapping leads to more efficient learning methodologies, thus laying the groundwork for more complex algorithms that combine TD methods with policy control.

Q-Learning

Q-Learning is a pivotal algorithm in model-free RL that seeks to learn the value of action-state pairs without requiring a model of the environment, cf. [144]. Essentially, Q-learning uses the Bellman equation as a basis to iteratively update

[2]please refer to [128] for an introduction to bootstrapping.

Q-values based on experience, enabling model-free learning. Within this framework, ”Q” symbolizes the action’s quality, indicating the action’s effectiveness in securing future rewards. Usually, future rewards are discounted by a factor γ per timestep [100]. Using this, the future discounted return at time t can be defined as

$$R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}, \tag{2.9}$$

where T is the timestep at which the game terminates.

Q-Learning is designed to learn the value $Q^*(s_t, a_t)$ of an action a_t in a specific state s_t to maximize the total expected reward over all successive steps $s_{t+1}, s_{t+2}, ...,$ starting from the current state s_t. The previously depicted Bellman equation plays a fundamental role in calculating the optimal action-value function. This principle emerges from the rationale that if we were to know the optimal value $Q^*(s_{t+1}, a_{t+1})$ for every possible action a_{t+1} in the subsequent state s_{t+1}, the best course of action would be to choose a_{t+1} that maximizes the expected return of $r+\gamma Q^*(s_{t+1}, a_{t+1})$, where r_{t+1} is the immediate reward and γ is the discount factor for future rewards. This concept is mathematically represented as:

$$Q^*(s_t, a_t) = \mathbb{E}\left[r_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1})|s_t, a_t\right] \tag{2.10}$$

Q-learning operationalizes this concept by applying an update rule to the Q-values based on the observed experience. The Q-value of a state-action pair is updated as follows:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right] \tag{2.11}$$

As experiences are gathered, basic Q-learning uses this equation to iteratively update a Q-table, essentially mapping states, values, and actions. This update is driven by the TD error, the discrepancy between the current estimate of $Q^*(s_t, a_t)$ and the new estimate formed from the newly observed reward and the maximum predicted Q-value of the next state.

Q-learning is an off-policy algorithm that learns the optimal policy’s value independently of the agent’s current actions, facilitating exploration without jeopardizing the acquisition of optimal actions. Off-policy algorithms allow learning from actions outside the current policy, enabling efficient exploration and exploitation balance but may lead to convergence issues if not carefully managed [129].

The algorithm’s capacity to identify the optimal policy without necessitating an environmental model is a pivotal advantage, as it allows agents to learn effective strategies directly from interactions, circumventing the need for complex representations or simulations of the environment [109]. furthermore, its straightforward implementation contributes to its widespread use, facilitating rapid deployment in various applications without extensive customization [4]. However, Q-learning's performance diminishes in environments characterized by large state spaces. An increase in potential state-action pairs demands substantial computational resources and memory for the Q-table [94]. Additionally, the effectiveness of Q-learning hinges on accumulating a vast amount of experience. This necessitates extended periods of interaction with the environment, which can be impractical in complex or dynamically changing scenarios [103].

State-Action-Reward-State-Action

The on-policy algorithm State-Action-Reward-State-Action (SARSA), aims to update the Q-value for policy improvement, just like Q-learning. Contrary to the off-policy method Q-learning, where the Q-value update is based on the maximum estimated future reward, however, it does so based on the policy's actual choices, incorporating the estimated Q-value of the next state-action pair chosen according to the current policy.

The SARSA algorithm updates the Q-value for a state-action pair (s_t, a_t) using the formula:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)], \tag{2.12}$$

where:

- $Q(s_t, a_t)$ is the current estimate of the Q-value for state s_t and action a_t,
- α is the learning rate,
- r_{t+1} is the reward received after taking action a_t in state s_t,
- γ is the discount factor, representing the importance of future rewards,
- s_{t+1} is the new state after taking action a,
- a_{t+1} is the next action taken, chosen according to the current policy,
- $Q(s_{t+1}, a_{t+1})$ is the estimated Q-value for the next state-action pair (s_{t+1}, a_{t+1}).

This formula highlights the difference between Sarsa and Q-learning. SARSA calculates the TD using both the current and subsequent state-action pairs, thus needing to know the future action a_{t+1} for the update process.

Going back to the grid world example, where an agent moves to reach a goal state from a starting position. Using SARSA, the agent updates its Q-values based on the policy it follows, for example, an ϵ-greedy policy where it mostly chooses the best-known action but occasionally tries a random action. The Q-values guide the agent through the grid, learning from each step and its outcome, adjusting its path based on both immediate rewards and estimated future rewards of subsequent actions.

SARSA, being an on-policy learning method, enables the algorithm to potentially lead to safer or more conservative learning outcomes, effectively avoiding risky moves that might seem optimal in theoretical scenarios but are hazardous in real-world applications [129]. It moreover exhibits a high degree of flexibility with regard to the policies it can employ. This adaptability allows it to be effectively integrated into a variety of scenarios, thus broadening its applicability across diverse RL tasks [129]. On the other hand, the requirement for SARSA to explore and learn the policy in a simultaneous manner can sometimes result in slower convergence rates, especially when compared to off-policy methods such as Q-learning [129]. This characteristic may lead to inefficiencies in environments where rapid learning of optimal policies is crucial. Furthermore, SARSA may not always find the most optimal policy as efficiently as some off-policy methods. This is particularly evident in environments characterized by high variability or risk, where the exploration necessary for on-policy learning can result in less efficient policy optimization [140].

Policy Gradient

Policy Gradient methods represent a different approach to the RL problem, where the focus is on directly learning the policy π— a mapping from states s to actions a — without relying on a value function $V(s)$ [131]. By adjusting the parameters of the policy in the direction of greater rewards (the gradient), the agent iteratively improves its behavior.

The foundational concept of Policy Gradient methods revolves around the policy function, $\pi_\theta(A_t = a|S_t = s) \forall A_t \in A(s), S_t \in S$, representing the probability of taking action a in state s, parameterized by θ [131]. Unlike value-based methods, which first learn a value function and derive a policy from it, Policy Gradient methods learn the policy function directly by adjusting the parameters θ to maximize the expected return, $J(\theta)$, from the start state. This is often expressed as:

$$J(\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta}[R(\tau)], \tag{2.13}$$

where $R(\tau)$ is the total discounted reward of a trajectory τ and $\mathbb{E}_{\tau \sim \pi_\theta}$ denotes the expectation taken over the distribution of all possible trajectories τ generated by following the policy π_θ. The Policy Gradient theorem provides a way to compute the gradient of $J(\theta)$ as:

$$\nabla_\theta J(\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R_t \right], \tag{2.14}$$

where R_t is the discounted return from time t. This gradient can then be used to update the parameters θ using gradient ascent, thereby improving the policy [118]. The policy gradient algorithm then performs a stochastic gradient ascent:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi_{\theta_k}), \tag{2.15}$$

where α represents the learning rate.

The Policy Gradient's direct way of optimizing the policy can lead to better convergence properties for problems with high-dimensional action spaces or continuous action spaces [126]. Additionally, the capability to learn stochastic policies, allows these methods to naturally represent and learn stochastic policies, which can be advantageous in certain environments [129]. On the downside, the high variance of gradient estimates can make training less stable and require more samples to converge [117]. This is coupled with a sensitivity to hyperparameter tuning, such as learning rate α and discount factor γ, which can significantly affect performance [117].

2.1.6 Deep Reinforcement Learning Algorithms

Traditional RL methods excel in environments characterized by discrete and finite state and action spaces, however, their applicability becomes constrained when faced with high-dimensional spaces, primarily due to the curse of dimensionality, which leads to an exponential increase in the number of state-action pairs [13]. To circumvent the limitations inherent in traditional RL, DRL integrates deep neural networks.[3]. Through the usage of deep learning, DRL simplifies complex environments, allowing agents to make decisions in situations with a lot of information or a wide range of possible actions [100, 125]. In the following, I will proceed to examine key DRL algorithms instrumental in advancing the field, highlighting their distinct methodologies and contributions to overcoming complex challenges through the fusion of deep learning and RL. The subsequent selection includes Deep Q-Networks (DQN) [101] and Proximal Policy Optimization (PPO) [119]

[3]This dissertation assumes a foundational understanding of neural networks. For readers seeking an introduction or a more comprehensive understanding, the following sources are recommended [57, 85].

with the intention of featuring an algorithm for both sides of the model-free taxonomy (q-learning and policy optimization)[4]. The algorithms also support the use of discrete action spaces, reducing the number of possible error sources.

Deep Q-Learning

The limitation of traditional tabular Q-learning is its inability to scale to problems with complex or continuous state spaces. Tabular Q-learning maintains a lookup table to store Q-values for every possible state-action pair. However, as the complexity of the environment increases, so does the dimensionality of this table, making tabular methods impractical. To manage more complex environments, the enhancement of Q-learning through the integration of neural networks, known as Deep Q-Learning or Deep Q-Networks (DQN), can offer significant improvements [101]. DQN facilitates the approximation of Q-values for a vast number of state-action pairs that would be infeasible to store and compute explicitly.

In DQNs, the neural network is parameterized by weights and biases collectively denoted by θ. The network is tasked with estimating Q-values, denoted as $Q(s, a|\theta)$. A pivotal enhancement introduced in DQN is the utilization of a *target network* and the concept of *experience replay*. The experience replay describes a process in which the DQN stores the agent's experiences at each timestep, $(s_t, a_t, r_{t+1}, s_{t+1})$, in a dataset D, known as the replay buffer. It then samples mini-batches uniformly at random from this buffer $(U(D))$ to perform updates. This method breaks the correlation between consecutive learning updates, significantly improving learning stability and efficiency by reusing past experiences. The target network estimates the Q-value of the next state-action pair, denoted by $Q(s_{t+1}, a_{t+1}|\theta_i^-)$. The target network represents a copy of the Q-network, held constant to stabilize the target Q-values during the learning updates. Periodically, the weights of the Q-network θ_i are copied to the target network to slowly incorporate parameter updates, reducing the risk of divergence or oscillations in the learning process due to the moving target problem. The loss function, central to the optimization of the DQN, is defined as:

$$L(\theta_i) = \mathbb{E}_{(s_t,a_t,r_{t+1},s_{t+1})\sim U(D)}\left[\left(r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}|\theta_i^-) - Q(s_t, a_t|\theta_i)\right)^2\right] \tag{2.16}$$

The loss function $L(\theta_i)$ aims to minimize the difference between the predicted Q-values by the current network parameters θ_i and the target Q-values, which are

[4]Please refer to [4] for a comprehensive overview of the model-free taxonomy.

calculated using the target network's parameters θ_i^-. This difference, squared, constitutes the TD error, reflecting the discrepancy between the current estimate and the optimal target estimate. Parameter updates are then applied according to the gradient of this loss function with respect to θ, leading to:

$$\theta_{i+1} = \theta_i + \alpha \nabla_\theta L(\theta_i) \tag{2.17}$$

This process enables the neural network model to effectively generalize across the state and action space. Unlike tabular approaches, where the scalability is directly limited by the cardinality of $S \times A$, the parameterization through θ in DQN allows for generalization, significantly reducing the dimensionality challenge. Nonetheless, the methodology may lead to oscillations or divergence in Q-value updates due to the concurrent use of the network for generating and updating target Q-values. Techniques such as experience replay and the use of fixed target networks aim to mitigate these challenges, thus stabilizing the learning process within the Deep Q-Learning framework.

DQNs are highly popular among computer scientists as they are comparatively simple to hyperparametrize, have a high sample efficiency, and proved successful in achieving superhuman performances in games [101]. This success underscores DQNs' capacity to handle complex decision-making tasks, significantly extending the applicability of RL. Just like Q-learning, DQNs operate in an offline manner, however, the offline nature of DQNs necessitates careful consideration of the exploration versus exploitation balance and the design of the replay buffer to ensure effective learning [66].

Proximal Policy Optimization

PPO emerges as a refinement of policy gradient methods, specifically designed to address their computational complexity and instability issues. PPO strikes a balance between ease of implementation, sample efficiency, and robustness, making it particularly well-suited for a wide array of RL problems, including those encountered in high-dimensional environments [118].

Central to PPO's approach is the idea of optimizing a surrogate objective function while ensuring that the new policy does not deviate too far from the old policy. This is achieved through the use of a clipped probability ratio, which bounds the amount of change introduced in the policy update. The objective function of PPO is expressed as:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right], \tag{2.18}$$

where:

- $\hat{\mathbb{E}}_t$ represents the empirical expectation over a batch of samples at timestep t, emphasizing the method's reliance on sampled data to approximate the expected value.

- $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio, indicating how the current policy's probability of selecting action a_t in state s_t compares to the probability under the old policy.

- $\hat{A}_t$ is an estimator of the advantage function at timestep t, measuring the benefit of choosing action a_t over other possible actions in terms of expected future rewards.

- The clip function bounds $r_t(\theta)$ within the range $[1-\epsilon, 1+\epsilon]$, with ϵ being a small positive value (hyperparameter) that determines how much the policy is allowed to change at each update.

This clipped objective function helps to prevent the policy from changing too dramatically, which could destabilize the learning process. By taking the minimum between the unclipped and clipped objective, PPO encourages policy improvement while ensuring the updates remain within a safe margin defined by ϵ. PPO's introduction of clipping and the empirical expectation calculation reflects a practical approach to implementing policy gradient methods, particularly suitable for environments with large or continuous action spaces. It balances the exploration of new policies with the stability of incremental learning, making it one of the most widely used algorithms in DRL today.

PPO's clipped objective function mitigates the risk of excessively large policy updates, a common pitfall in earlier policy gradient methods that could lead to destructive large steps in the policy space [119]. The algorithm facilitates multiple epochs of minibatch updates using the same trajectory data, enhancing sample efficiency—a critical advantage in complex environments [119, 64]. This efficiency is complemented by PPO's flexibility to work with both continuous and discrete action spaces, broadening its applicability across various DRL scenarios. Compared to its predecessors and contemporaries, such as Trust Region Policy Optimization (TRPO)[5], PPO offers a more straightforward implementation without compromising performance. It eliminates the need for a complicated second-order optimization by approximating the trust region constraint with a simpler clipping mechanism [119]. Despite its advantages, PPO is not without challenges. Fine-tuning its hyperparameters, like the learning rate α and clipping parameter ϵ, remains crucial for achieving optimal performance [65].

[5]Please refer to [116] for an in-depth introduction to TRPO.

2.2 Dynamic Pricing

Dynamic pricing, also referred to as real-time pricing or demand pricing, is an adaptive pricing strategy whereby businesses set flexible prices for products or services based on current market demands. This approach leverages advanced data analytics and machine learning algorithms to adjust prices in response to a variety of factors including changes in supply and demand, competitor pricing, and customer behavior patterns. The underlying premise is that by dynamically adjusting prices, businesses can maximize their revenue and profit margins while maintaining competitiveness in the market (cf. [21]). Research indicates that dynamic pricing can lead to an increase in sales volume and profit margins. Dynamic pricing strategies, when properly implemented, can improve profitability by enabling sellers to adjust prices based on the real-time balance of supply and demand [45]. Due to the rising adoption of RL throughout the last decade, dynamic pricing algorithms have shifted from static heuristics to AI-driven software as they outperform them in terms of average daily profits [83]. Studies have already shown that in certain markets it is unreasonable to conduct business at a profitable level lacking this technology, as algorithmic price setting frequencies create a significant competitive edge [12].

2.2.1 Dynamic Pricing in E-Commerce

The rise in online shopping worldwide has helped E-Commerce grow significantly, making up 14.1% of all sales and reaching \$3.5 trillion in 2019 [34]. This trend is further bolstered by advancements in information technologies and the expansion of transportation networks, facilitating increased availability of goods worldwide. Consequently, local suppliers find themselves competing on a global scale, necessitating adaptive strategies to stay competitive. With the advent of the big data revolution and the proliferation of cloud computing [78], companies are increasingly leveraging data analytics to inform and potentially automate decision-making processes. There is a discernible shift towards data-driven insights and over-reliance on intuitive judgment among sellers, leading to the processing of extensive data to glean customer preferences, disseminate pertinent information, and identify prevailing market trends to capture a larger share of consumers' spending [38].

This data-centric paradigm is driving the adoption of more frequent and targeted price adjustments aimed at maximizing revenue while maintaining consumer satisfaction. As such, sellers make use of dynamic pricing, also known as demand-based pricing. This pricing strategy aims to adjust prices in real-time in response to market conditions to maximize profitability by optimizing the price at which goods or services are sold [28]. Dynamic pricing is based on the economic principle of supply and demand. When demand is high, prices increase to balance the demand and supply. Conversely, when demand is low, prices decrease to encourage purchase [132].

This principle offers several benefits. It allows businesses to respond quickly to changes in the market, making them more competitive [150]. It also enables businesses to manage inventory more effectively, reducing waste and increasing efficiency [91]. Furthermore, dynamic pricing can lead to increased revenue and profitability [20]. Nevertheless, dynamic pricing has been criticized for potentially leading to price discrimination [54]. Some consumers may end up paying more for the same product or service based on factors such as their location or the time of purchase [153]. Therefore, businesses must implement dynamic pricing strategies carefully to avoid potential backlash [53].

In the E-Commerce sector, the comprehensive implementation of dynamic pricing strategies has become virtually indispensable for maintaining competitiveness [55]. As a result, prices no longer merely mirror fluctuations in supply and demand but also encapsulate evolving opportunity costs in the face of demand uncertainty and scarcity. The cost associated with selling a unit of inventory at present is contingent upon a firm's potential to sell it in the future. Moreover, demand may exhibit predictable temporal variations. For instance, if consumers with a high willingness to pay tend to arrive later, firms may have an incentive to conserve inventory. In all the aforementioned scenarios, prices may be adjusted in response to competitive interactions. However, a significant portion of the theoretical and empirical literature on dynamic pricing in perishable goods markets has entirely abstracted from competition. This paper introduces a framework to study dynamic price competition. We explore how dynamic price competition can detrimentally impact market efficiency and how alternative pricing mechanisms can enhance overall welfare [17].

2.2.2 Deep Reinforcement Learning in Dynamic Pricing

Dynamic RL-based pricing strategies supersede static ones in terms of average daily profits [83]. As 27 percent of the respondents of a 2017 study by KPMG identified price or promotion as the factors that are most likely to influence their decision regarding which product or brand to buy online [82], it is a logical consequence that successful companies (such as Amazon [29]) base their decisions on these algorithms to learn from and react to their competitor's pricing policies as well as to adjust to external factors, such as a transformation of demand or product innovations [49].

In various studies, researchers have explored (D)RL algorithms to enhance dynamic pricing approaches. As such, Lu et al. [93] explored the application of

RL within the context of a hierarchical smart-grid system. By implementing Q-learning, the service provider can determine the retail electricity price adaptively, considering the uncertainty of customers' load demand profiles and the flexibility of wholesale electricity prices. Experimental results highlight improved service provider profitability, reduced customer energy costs, balanced energy supply and demand in the electricity market, as well as improved reliability of electric power systems. Zhang et al. [151] confirm these findings within a review of the application of DL, RL, and DRL in smart grids. They discuss how these AI techniques, especially DRL, can enhance prediction accuracy, anomaly detection, decision-making in smart grids, and DRL's ability to make strategic decisions in complex environments.

In an exploratory study, a researcher described the optimization of a dynamic congestion pricing system in conjunction with RL [11]. Based on highway congestion levels, an RL algorithm would dynamically adjust the price for a toll road. Although RL proved to be quite demanding for the proprietary traffic simulator used in the study, the authors concluded that it can inform basic feedback control methods.

Other research focused on simulating (online) markets. As such, Kastius and Schlosser [75] explore the use of DQN and SAC for dynamic pricing in various market models. The study finds that both algorithms produce reasonable results, with SAC performing better than DQN, and suggest that under certain conditions, RL algorithms can be induced into collusion without direct communication. Others present an end-to-end framework using DRL for dynamic pricing on E-Commerce platforms [90], contributing by showing significant performance improvements over manual pricing by extending the problem to continuous price sets and defining a novel reward function. Furthermore, Reza Refaei et al., [5] feature a novel hyper-parameter optimization method and demonstrate significant revenue improvement over benchmark methods in the context of online advertising.

2.2.3 Evidence of Deep Reinforcement Learning in E-Commerce

However, RL has not only been investigated in experimental setups. Evidence from major online retailers suggests that RL-based dynamic pricing, also known as pricing agents, is now a standard practice. A study of Germany's retail gas market identified a large-scale adoption of pricing algorithms since 2016 based on a catalog of potential characteristics. As a consequence, sellers reached margins above competitive levels. As their data indicates no initial effects, followed by an eventual convergence to high prices and margins, they infer that the algorithms were able to learn tacitly collusive strategies over time [12].

Brown and MacKay [22] tackle the issue from a different angle, examining the strategic use of pricing algorithms among five pharmacy firms with varying frequencies of price changes. Their findings suggest that firms employing high-frequency pricing algorithms not only adjust to market dynamics more effectively but also potentially gain a competitive edge, leading to increased price levels and affecting the overall market dynamics, including price dispersion and the impact of mergers on pricing strategies [22].

Researchers from Alibaba, one of the leading E-Commerce shops [9], address the problem of dynamic pricing on E-Commerce platforms using methods based on DRL [90]. The study demonstrates the superiority of employing a continuous pricing strategy over a traditional discrete approach, innovates with a new reward function named the difference of revenue conversion rates, and addresses the initial data scarcity challenge through pre-training on historical transactions. This DRL-based method surpasses traditional expert-driven pricing tactics, showcasing a breakthrough in implementing DRL for practical E-Commerce pricing scenarios.

Further evidence of DRL's impact on E-Commerce is highlighted through studies conducted on Amazon [29] and Bol.com [62]. The research on Amazon demonstrates how algorithmic pricing strategies, utilized by over 500 sellers, foster price competitiveness and volatility, emphasizing the significance of dynamic pricing in enhancing market presence [29]. Similarly, an empirical analysis on Bol.com, the largest online marketplace in the Netherlands and Belgium, reveals that algorithmic sellers adjust prices dynamically, influencing the Buy Box's price [62]. These findings underscore the widespread adoption of DRL-based pricing strategies across major online platforms, contributing to a more dynamic E-Commerce ecosystem.

2.3 Antitrust Law

This thesis, while primarily focused on AI, acknowledges the importance of incorporating foundational legal principles to interpret the results and discuss their legal implications accurately. This interdisciplinary approach ensures that our findings are not only technically robust but also applicable under Article 101 Treaty on the Functioning of the European Union (TFEU)[6] and Section 1 Act against Restraints of Competition (ARC) of the German competition law.

2.3.1 Mutual Agreement

The primary objective of antitrust law is to promote market competition and, by extension, safeguard the interests of consumers [77]. To achieve this, business en-

[6] For US law see [138, 59].

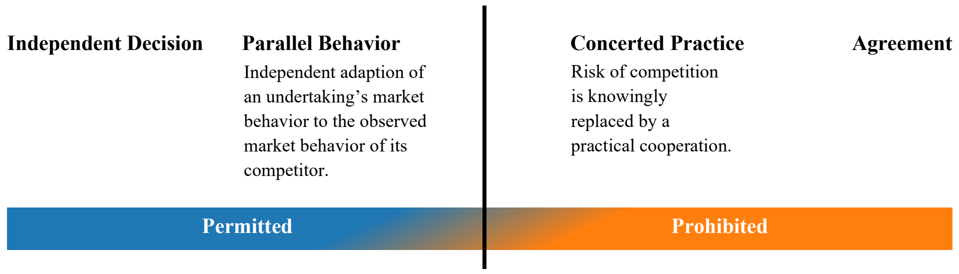

Independent Decision
Parallel Behavior
Independent adaption of an undertaking's market behavior to the observed market behavior of its competitor.
Concerted Practice
Risk of competition is knowingly replaced by a practical cooperation.
Agreement
Permitted
Prohibited

Figure 2.3: A simplified differentiation of parallel behavior and concerted practices.

tities, referred to as ”undertakings”, operating within the market must act independently from one another. This principle of independence mandates that any form of direct or indirect interaction among market players, which could potentially influence or reveal one's market strategy to a competitor, is strictly prohibited [47]. Such interactions are considered harmful as they might distort market conditions from what would naturally arise from uninfluenced competition.

Article 101 of the TFEU and Section 1 of the ARC explicitly forbid any form of agreement or coordinated behavior among businesses that could restrict market competition [47, 23]. Undertakings shall independently decide over their market behavior and must not coordinate with their competitors. This is known as the ”requirement of independence”. This includes decisions made by groups of companies and any joint actions that could lead to competitive constraints. Essentially, these rules draw a clear line between acceptable independent actions and illegal cooperative tactics. For an action to fall under the prohibition of Article 101 TFEU and Section 1 ARC, it must involve companies coming together to agree on a mutual course of action in the market (cf. Figure 2.3). Both European and German competition law distinguish three possible conducts of infringing the cartel prohibition:

1. Agreements between undertakings,
2. Decisions by associations of undertakings,
3. Concerted practices.

Agreements Between Undertakings

An agreement between undertakings encompasses any form of agreement, arrangement, or concerted action between two or more undertakings that have the object or effect of preventing, restricting, or distorting competition within the internal market. These agreements can be formal or informal, written or oral, and include both horizontal agreements (between competitors) and vertical agreements (between companies at different levels of the supply chain, such as manufacturers and distributors). Examples include price-fixing, market-sharing, and production limitation agreements.

The prohibition is aimed at agreements that have an anti-competitive object (i.e., their purpose is to restrict competition) or an anti-competitive effect (i.e., they may not intend to restrict competition but do have that effect). The assessment of these agreements often involves analyzing their context and the competitive structure of the market.

Decisions by Associations of Undertakings

This category targets decisions made by associations or collective groups of businesses, such as trade associations or professional organizations. Similar to agreements between undertakings, these decisions are prohibited if they have the object or effect of restricting competition. Examples include collective price-setting, adoption of standards that exclude certain competitors, or coordinated market-sharing.

Decisions by associations can significantly impact market competition, as they can influence the behavior of a number of companies within an industry. The scrutiny under this provision ensures that collective actions by businesses through their associations do not undermine competitive dynamics.

Concerted Practices

Concerted practices refer to forms of coordination between undertakings that, without having reached the stage of a formal agreement, knowingly substitute practical cooperation between them for the risks of competition. This could involve activities where companies align their behavior on the market, such as by sharing sensitive market information, that lead to a condition of reduced competition without the need for an explicit agreement.

The concept of a concerted practice is somewhat broader and more nuanced than explicit agreements or decisions, capturing situations where companies may not explicitly agree to engage in anti-competitive behavior but engage in coordinated actions that have a similar effect. Identifying concerted practices often involves analyzing patterns of market behavior and communication between companies to infer collusion.

2.3.2 Pricing Agents and Antitrust Law

In a proposal for an Artificial Intelligence Act, the European Commission emphasizes the importance of the safety and lawfulness of AI systems, of legal certainty concerning AI, the governance and effective enforcement of existing law on fundamental rights, and the installation of safety requirements [32]. In line with these goals, AI price policies must oblige competition law just as prices set by humans. Essentially, undertakings must ensure that their pricing agent refrains from any form of communication or interaction that could affect a competitor's business strategy or reveal its own plans and strategies to a competitor. The purpose of this rule is to ensure that the competitive environment remains natural and is not distorted by agreements or shared plans among competitors that could lead to unnatural market conditions [43].

The independently chosen intelligent adaption of an undertaking's market behavior to the observed market behavior of its competitors (generally) is permitted. Drawing a clear line between the adaption of an observed market behavior and a conduct through which competition knowingly is replaced by practical cooperation and therefore constitutes a concerted practice within the meaning of Article 101 (1) TFEU is often difficult and sometimes even impossible. Especially in transparent markets with few market participants, the market outcome of collusion can often hardly be traced back to be (or not to be) the product of a concerted practice (cf. petrol station market). While collusion, or companies working together in ways that harm consumers, innovation, and economic growth, is bad for the economy and undesired from a welfare perspective, it's challenging to legally hold anyone accountable for such market outcomes from a theoretical standpoint. The rationale behind this is that the law typically requires specific actions or agreements to impose liability, not just harmful market effects. [145].

Our goal is to disclose whether a certain sequence of actions or a specific pattern can be identified as a situation in which the uncertainty about the competitor's next moves is replaced by a practical cooperation. The rise of self-learning algorithms in the market, which can analyze vast amounts of important data and respond to price changes very quickly, may increase market transparency to such an extent that it becomes difficult to tell if market outcomes are the result of legitimate competitive strategies or illegal collusion. These algorithms' ability to monitor the market so closely and act so frequently makes it challenging to distinguish between companies independently making similar decisions and those engaging in coordinated, anti-competitive practices.

However, when no such coordination can be identified, the collusive market outcome generally must be considered legally neutral. Although collusion can be detrimental to consumers, innovation and economic growth are therefore undesirable from a welfare economic point of view, the difficulty from a dogmatic perspective is that legal responsibility cannot easily be attached to a market outcome as such [31]. This evokes three major legal questions. First, does the agent's behavior constitute a minimum degree of coordination and thus violate the cartel prohibition under existing German and European competition law? Second, do the existing rules readily fit an algorithmic conduct or do we face a regulatory gap within the cartel prohibition? Third, should we expand cartel law to encompass algorithmic collusion?

2.4 Economics

In this section, we will delve into the fundamental concepts of economics that are central to discussions on market structures, demand and supply, pricing strategies, and the strategic interactions among firms. We will start with an introduction to market structures, exploring the differences between perfect and imperfect markets, and then move on to an analysis of demand, supply, and price determination. Furthermore, we will discuss the implications of supracompetitive pricing, the application of game theory to strategic interactions, and the specific models of Bertrand and Cournot competition. Finally, we will address the importance of ensuring consumer welfare in the context of antitrust laws and economic efficiency.[7]

2.4.1 Introduction to Market Structures

In the general understanding, a market is typically seen as a physical location where goods are purchased and sold. However, economists define a market as a collection of buyers and sellers engaged in transactions involving a specific type of goods or services. The demand for a product is determined by the buyers, while the supply of the product is determined by the sellers [96].

Market structures are fundamentally categorized based on the number of firms in the market, the nature of the product being offered, entry and exit barriers, and the degree of information symmetry among participants [96]. As displayed in 2.4 research primarily differentiates perfect and imperfect market forms. Perfect competition embodies a market with numerous small firms, homogeneous products, no barriers to entry or exit, and perfect information. In such markets, no single entity, neither buyer nor seller, has the market power to influence prices, thus acting as ”price-takers”. Price-takers must accept prevailing prices in a market, lacking the ability to influence market prices independently. Imperfect markets conversely, are

[7]you can find a thorough introduction to Economics in [96, 84].

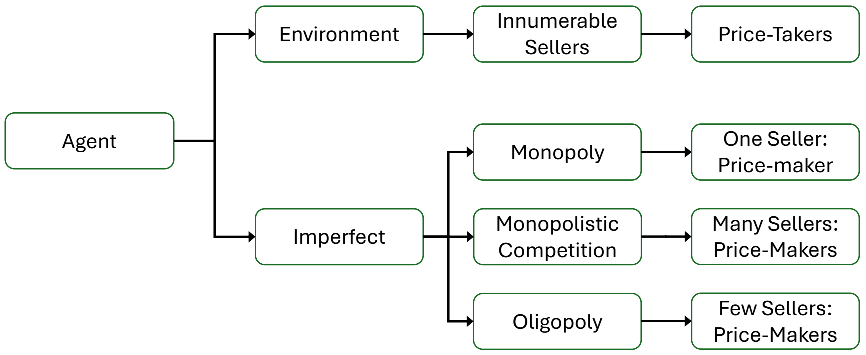


Figure 2.4: Traditional types of competition.

characterized by single or few firms dominating the market, monopolistic competition, and asymmetric information. Imperfect competition is also characterized by barriers to entry and exit, which prevent the market from adjusting to new entrants or exits smoothly. Imperfection generally leads to sellers acting as ”price-makers”. However, the degree of freedom in their price selection is influenced by the specific characteristics of the market structure and the level of competition within it [84].

In the following thesis, our focus will primarily be on oligopolies and monopolistic competition, as these market structures present unique challenges and opportunities for deploying pricing strategies through DRL agents. Oligopolies, a market characterized by few dominating firms, provide a unique setting to explore strategic pricing behaviors, including tacit collusion, price wars, and the impact of market entry or exit. This market structure reflects many modern industries, such as telecommunications, airlines, and pharmaceuticals, where the understanding of pricing dynamics can have significant economic and regulatory implications [122, 26]. Monopolistic competition, defined by many firms selling differentiated products, allows for an investigation of how product differentiation influences pricing strategies. It offers insights into how firms balance between pricing for market penetration and pricing for profit maximization. This market structure is particularly relevant for sectors with high levels of product innovation and marketing, such as consumer goods and technology markets [137, 80].

2.4.2 Demand, Supply, and Price Determination

In economics literature, the market is observed from different perspectives, the buyers and sellers. From a seller's perspective, we define *The law of demand*. This law posits that when the price of a good rises, the quantity demanded of the good falls, and vice versa. Conversely, from a buyer's perspective, the research describes

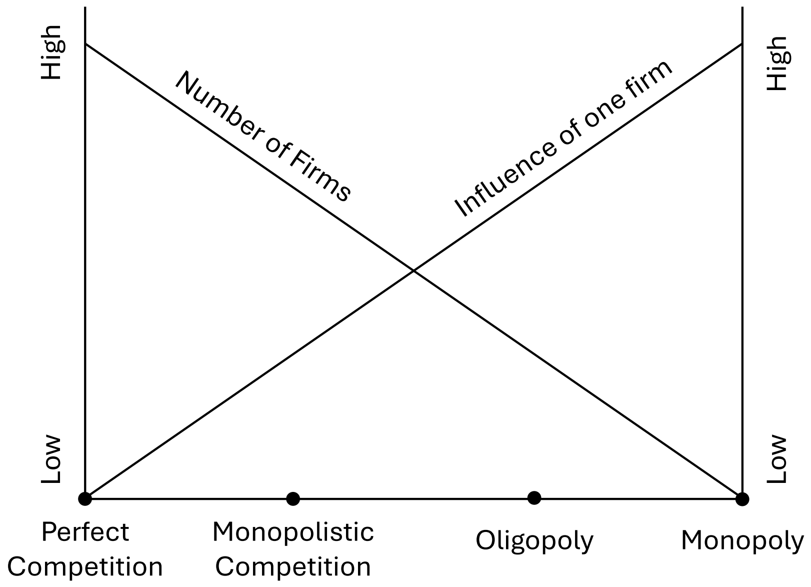


Figure 2.5: Classification of market structures based on the number and influence of firms on a single market.

the law of supply, suggesting that a higher price will encourage producers to supply a greater quantity of the good. The price of a product or service where the quantity demanded by consumers equals the quantity supplied by producers is known as the market equilibrium (cf. Figure 2.6). This equilibrium price ensures that the market clears, with no excess supply or demand [68].

On a price-demand curve, a monopolist can set the price above the marginal cost, leading to an allocative inefficiency, causing a deadweight loss to society. This pricing strategy will conversely result in a lower quantity of goods. In contrast, in a market characterized by perfect competition, prices tend to align more closely with the marginal cost of production. The large number of firms and the homogeneity of the products ensure that no single firm has significant market power to influence prices. Here, the equilibrium price is determined at the intersection of the market supply and demand curves, where the quantity supplied equals the quantity demanded (cf. Figure 2.6). This results in a lower price and higher quantity of goods sold compared to a monopolistic market, thereby maximizing consumer and producer surplus without generating significant deadweight loss [74].

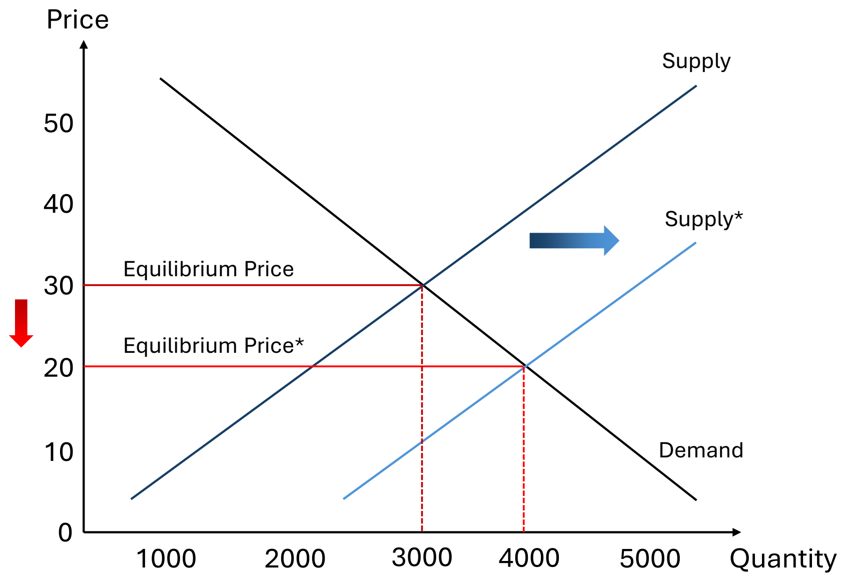


Figure 2.6: Formation of the equilibrium price on a market based on demand and supply.

2.4.3 Supracompetitive Pricing

Supracompetitive pricing refers to the strategy of setting prices above what would be expected in a perfectly competitive market where prices typically reflect the marginal cost of production. This pricing strategy is commonly observed in market conditions characterized by limited competition, such as oligopolies or monopolistic markets, where firms possess sufficient market power to influence prices to their advantage (cf. [98] and [53]).

In economic terms, supracompetitive prices exceed the level necessary to cover all costs, including the normal return on investment. Such pricing can lead to higher profit margins for the firms involved but often at the expense of consumer welfare and market efficiency. This phenomenon is particularly relevant in discussions of market power and regulatory responses. For instance, when a small number of firms control a significant portion of the market, they may tacitly or explicitly collude to maintain prices at a supracompetitive level, thereby restraining trade and reducing total welfare in the market. Supracompetitive pricing can distort market signals, resource allocation, and the competitive landscape, potentially leading to market failures.

2.4.4 Game Theory and Strategic Interaction

To fully understand how AI pricing agents interact in market environments, especially in oligopoly and monopolistic competition, it is necessary to introduce foundational concepts of game theory. Central to game theory is the concept of Nash equilibrium, a situation in which no player can benefit by unilaterally changing their strategy, given the strategies of all other players remain constant. In the context of pricing strategies, Nash equilibrium represents a state where each firm's pricing decision is optimal, considering the pricing decisions of its competitors. Analyzing these interactions helps illuminate the conditions under which pricing agents might arrive at competitive equilibria or, contrarily, collude to set prices above the competitive level [37, 127][8].

Game theory furthermore differentiates between repeated games and single-shot games. Repeated games consider the possibility of interactions occurring over multiple periods, allowing for the study of strategies that evolve based on past actions [56]. This framework lends itself to analyzing the potential for tacit collusion among pricing agents. When firms interact repeatedly, they may sustain cooperation implicitly, avoiding overt collusion but achieving similar outcomes by learning and adapting to each other's pricing strategies over time. The anticipation of future interactions can thus foster a cooperative equilibrium, even in the absence of formal agreements, as long as the benefits of maintaining cooperation outweigh the short-term gains from deviating.[9]

2.4.5 Bertrand Competition

In a Bertrand competition model firms use their prices to compete for a homogeneous product. In this model, each firm independently chooses its price, assuming that the prices of its competitors remain constant. Consumers, facing no product differentiation aside from price, will purchase from the firm offering the lowest price [15].

A Bertrand Competition model with identical products and no capacity constraints can be solved using game theory. If Firm 1 and Firm 2 have the same marginal cost c, and Firm 1 sets a price $P_1 > c$, Firm 2 can set a price P_2 such that $c < P_2 < P_1$ to capture the market. Hence, rational behavior leads both firms to set $P = c$, resulting in a Nash equilibrium of $P_1 = P_2 = c$. This equilibrium is derived under the premise that if any firm sets a price above the marginal cost, a

[8]Please refer to [56] for an in-depth introduction to game theory

[9][33] and [135] investigate how repeated interactions and the prospect of future engagement can incentivize firms to adopt cooperative strategies, facilitating outcomes that mirror those of formal collusive agreements without explicit coordination.

competitor can slightly undercut this price, essentially capturing the entire market. Thus, firms earn zero economic profit, similar to perfect competition outcomes, despite a smaller number of firms in the market.

While this model underscores the potential intensity of price competition, it is often criticized for its assumption of perfect substitutability. As such, the Bertrand equilibrium may be less realistic in markets where products are differentiated or where strategic pricing involves more complex considerations beyond undercutting competitors' prices [98].

2.4.6 Cournot Competition

Cournot competition investigates a setting in which firms compete on quantities rather than prices. Each firm decides its production quantity, assuming its competitors' quantities are fixed. The market price is determined by the aggregate supply through a demand function. Cournot competition is relevant in markets where firms exert control over production levels but accept the market price as determined by total industry output [98].

To determine the equilibrium quantities in a Cournot duopoly mathematically, one considers two firms facing a linear demand function $Q = a - bP$, where Q represents the total quantity demanded, P is the price of the good, a and b are parameters determining the position and slope of the demand curve, respectively. Given constant marginal costs c for both firms, the inverse demand function, which relates price to the total quantity supplied by both firms, is derived as $P = \frac{a}{b} - \frac{Q}{b}$. The profit for Firm 1 (π_1) can be expressed as the difference between total revenue and total cost, or $\pi_1 = (P - c)q_1$, where q_1 is the quantity produced by Firm 1. Substituting P from the inverse demand function gives $\pi_1 = \left(\frac{a}{b} - \frac{(q_1+q_2)}{b} - c\right) q_1$, where q_2 is the quantity produced by Firm 2. To find the equilibrium quantity for Firm 1, we take the derivative of π_1 with respect to q_1, set it to zero, and solve for q_1 as a function of q_2. This results in Firm 1's reaction function, which shows how Firm 1 optimally adjusts its output in response to Firm 2's output level. Firm 2's equilibrium quantity and reaction function are derived symmetrically. The Cournot Nash equilibrium occurs at the point where these reaction functions intersect, meaning each firm's output level is the best response to the other firm's output level.

While the Cournot model is useful in analyzing markets where firms compete in quantities, and it incorporates the strategic interdependence of firms' decisions, it is limited by its assumption that firms choose quantities simultaneously, without considering potential reactions from competitors in real time. This assumption does not hold in markets where firms can quickly adjust their production in response to competitors [98].

2.4.7 Ensuring Consumer Welfare

As highlighted, antitrust laws are designed to preserve competitive independence among firms, prohibiting actions that undermine this principle. To establish a violation of these laws, it is often necessary to demonstrate evidence of collusion or other forms of unlawful cooperation between competitors. Economic theories and analytical tools enable investigators to scrutinize market outcomes, such as pricing patterns, to discern whether they are the result of competitive behavior or indicative of illegal coordination.

One indicator of potential anticompetitive behavior is supracompetitive pricing. This phenomenon occurs when prices are set significantly higher than what would be expected in a highly competitive market, which can be indicative of collusion among firms or an abuse of market power by a dominant player. Beyond merely suggesting anticompetitive behavior, supracompetitive pricing leads to a specific form of economic inefficiency known as allocative inefficiency [97].

Allocative inefficiency arises when the price of a good exceeds the marginal cost of its production, leading to reduced output from the level that would prevail in a competitive market (cf. 2.7. This misallocation results in a situation where the goods that are not produced and sold would have provided greater value to potential buyers than their production costs, representing a net loss to society. Essentially, the production and consumption of goods are not optimized, leading to a scenario where resources are not allocated to their highest-valued use [97].

Consider a market for e-books. In a competitive landscape, assume e-books are sold at €5 each, closely aligning with the cost of production plus a nominal profit margin. Now, if a single firm gains control over the majority of e-book rights it would essentially enable it to act as a monopolist. If that firm decides to price e-books at €10, several potential consumers who value the e-books between €5 and €10 are priced out of the market. These consumers would have purchased the e-books at a price reflective of their production costs plus a competitive profit (i.e., €5), but are unwilling or unable to do so at the monopolist's higher price point (i.e., €10). The reduction in output and the resultant higher price prevent these transactions, which would have been mutually beneficial under competitive conditions. The value that these foregone transactions would have added to society – representing consumer and producer surplus – is lost, epitomizing allocative inefficiency.

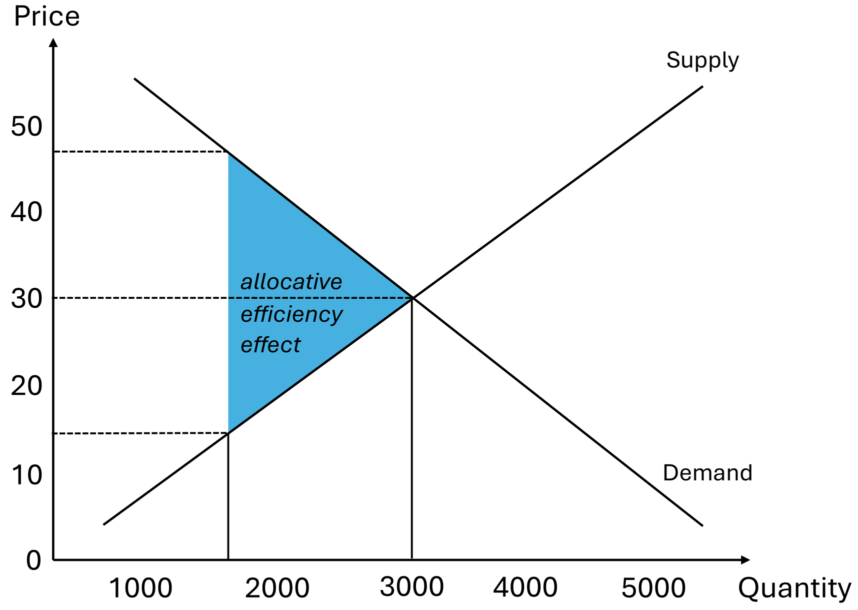


Figure 2.7: Allocative efficiency effect within a linear supply and demand model.

2.5 Related Work: Reinforcement Learning, Collaboration, and Collusion

In this section, we delve into the relevant literature surrounding the dynamics of cooperation and competition in RL. Hence, we provide a review of how RL agents develop cooperative behaviors in multi-agent systems and the mechanisms through which pricing algorithms might independently facilitate collusive behaviors in oligopolistic markets.

2.5.1 Cooperation in Reinforcement Learning

Particularly in the context of MAS, cooperation between RL agents is essential for tasks where individual agents must interact to strategically achieve a common goal. One of the primary methods to promote collaboration in RL is through the use of joint rewards. As such, scholars introduced an improved cooperative RL algorithm that utilizes joint rewards to ensure agents learn cooperative behaviors [149]. Another approach involves integrating social payoffs into the reward structure, which has been shown to facilitate cooperation in spatial prisoner's dilemma games [51]. Promoting cooperation among RL agents can also be facilitated through internal rewards [139, 152].

In recent years, researchers built on this by proposing collaborative DRL frameworks to address the heterogeneity among different learning tasks [87], by improving exploration by having agents share a common exploration goal [89], or by setting up coordinated challenges to search for superior performance in strategy games [111].

As highlighted by previous research, RL agents have demonstrated exceptional effectiveness in collaborative tasks. Building on this foundation, recent investigations have begun to explore the potential of RL agents to collaborate within competitive environments. These studies frequently utilize simple game scenarios to evaluate the agents' performance and interaction dynamics in competitive settings [133, 58, 52]. These studies have demonstrated an emergence of both competitive and collaborative behaviors between (D)RL agents, which were able to develop more robust strategies to maximize their benefits when pitted against adaptive agents. In a more recent study, researchers investigated the emergence of competitive or collaborative behaviors in a 3D simulated environment using a self-play approach. By comparing independently trained agents against those controlled by the same policy, revealing cooperative behaviors even in theoretically competitive scenarios [36].

2.5.2 Pricing Algorithms and Collusion

The study of pricing algorithms and their potential for collusion has garnered significant attention, particularly as advancements in AI and ML have enabled self-learning algorithms to adopt sophisticated strategies. This section delves into the mechanisms through which algorithms can independently achieve collusive behaviors. We review seminal and recent research that investigates these dynamics within oligopolistic markets, considering both theoretical frameworks and empirical studies. Key insights from game theory, RL, and economic simulations highlight the propensity for algorithm-driven tacit collusion and the conditions under which it arises.

Primarily legal scholars have commented on the possibility of self-learning algorithms to quickly learn to achieve a price-setting collaboration especially within oligopolies (e. g., [48, 49, 50]). With the power of modern hardware, AIs would be able to monitor the market in which they act, resulting in a rapidly arising tacit collusion. Researchers investigated the issue by creating game theory-like scenarios with the intention of pushing the agents towards a Nash equilibrium (e. g., [49, 143]). In essence, it seems to be "incredibly easy, if not inevitable" to achieve "such a tacitly collusive, profit-maximizing equilibrium" [120]. While collusion has been presumed to appear in enclosed MARL scenarios, scholars have neither studied how to spot the origin of collusion nor if competitors can apply tacit collusion by displacing others.

Researchers have investigated circumstances that can be juxtaposed with collusion between pricing agents, such as bidding processes [121, 39] or economy simulations [154]. However, the authors did not control for or induce communication or collaboration. To combat this shortcoming, scholars within the economics realm created oligopolistic models, particularly Cournot oligopolies (e. g., [143, 76, 124]) to show collusion between agents. Izquierdo and Izquierdo [70] show that simple iterative procedures, such as the win-continue, lose-reverse (WCLR) rule are able to achieve collusive outcomes. However, the results are not robust in terms of minor, independent perturbations in the firms' cost or profit functions. Similar results were achieved with basic Q-learning [143]. As a case in point, using a price-setting duopoly model with fixed production, in which two firms follow a Q-learning algorithm, Tesauro and Kephart [134] observed convergence to prices higher than the competitive level. Major takeaways from these studies are, that cooperation is more likely to occur in simplified, static environments with a homogeneous good and that communication is vital to achieve collusive outcomes, particularly when more than two firms operate in a market. Such results suggest that the ability to communicate could also be pivotal for algorithmic collusion to occur [120].

Other scholars investigated the issue by adding human participants [146], by specifically analyzing sizes of discrete action spaces [79], or by building custom, more elaborate scenarios [2]. Few researchers applied deep neural networks. The ones that did, were able to improve on Calvano et al. [24] by achieving a shorter learning time due to the usage of DQNs as well as reward averaging [67]. Others restricted algorithms to only memorize the periods when they do not exceed in terms of profits and ignored the ones when they outperform. Scholars, nevertheless, emphasize that "more efforts are needed in exploring other architectures of deep networks" [60]. Due to the characteristic learning behavior of RL-algorithms or AI, the quality of learning data significantly influences agents' propensity for collusion. To test this hypothesis, some scholars employed a simple upper confidence bound bandit algorithm to set a discrete number of prices [61]. Their outcome indicated that the prices are bound to the signal-to-noise ratio of their inputs, resulting in a supracompetitive state for less noisy input data and vice versa.

The current state of research is mainly simulation-based, with few scholars collecting empirical data. An investigation into Germany's retail gas market, conducted using a catalog of potential characteristics, identified widespread adoption of pricing algorithms since 2016. Consequently, sellers achieved margins above competitive levels. As their data indicates no initial effects, followed by an eventual convergence to high prices and margins, they infer that the algorithms were able to learn tacitly collusive strategies over time [12]. Brown and MacKay [22] tackle the issue from a different angle. The authors extract pricing data from five pharmacy firms with differing price changing frequencies [22]. Musolff [102] employs a dataset acquired from Amazon's buybox, an algorithmic pricing-heavy feature used by third-party sellers, to show that repricers have been able to avoid the competitive behavior by regular price raises.

Chapter 3

Investigating Collusion in a Toy Problem

While the previous chapter introduced the fundamental concepts of this thesis, this section delves into the results within a ”toy problem” scenario. A toy problem serves to demonstrate various problem-solving approaches. They are simplified, idealized systems that strip down a problem to its essential features, removing real-world complexities to clarify the underlying mechanics or principles.[1] In the context of studying collusion, we investigate a competitive MARL game simulation to observe the agents’ behavior in a controlled setting. The agents play a three-player version of Rock Paper Scissors (RPS). We aim to analyze the agents’ learning performances and potential collaboration strategies. Through a game theoretic approach, our study investigates how DRL agents make pricing decisions and how these decisions can lead to tacit collusion without explicit communication among the agents. The most significant observation is a demonstration of the ”stages of collusion”. The DRL agents’ action selection evolves throughout the runs, suggesting that these agents can independently learn to avoid competitive pricing strategies in favor of higher profits, thereby mimicking collusive behavior. Furthermore, we raise critical questions about the implications for market dynamics and regulatory frameworks.

The remainder of this paper is structured as follows. Section 3.1.1 introduces the term trustworthy AI in the context of reasonable machines and white-box AIs. We furthermore present a foundation of anti-competitive agreements and possi-

[1]While specific foundational texts in these areas, such as [142] for game theory, or [129] for AI and machine learning, do not necessarily introduce the term ”toy problem,” they employ the concept by discussing simplified models to explain intricate ideas.

ble overlaps betweenMARL, and related work in analyzing possible cooperation between RL agents. In section 3.2, we establish a case study that builds on DQNs-Agents to present the training process of collusion. Section 3.3 portrays the outcomes of the different learning sessions. Finally, we discuss our findings, highlight the boundaries of our study, and conclude with promising avenues for future research.

Parts of this work presented in this chapter have been published before in [114] and extended upon in [99].

3.1 Related Work

3.1.1 Trustworthy AI

As mentioned in Section 2.3, the European Commission emphasizes the importance of the safety and lawfulness of AI systems, legal certainty concerning AI, governance and effective enforcement of existing law on fundamental rights, and the installation of safety requirements to AI systems [32]. As AIs are inevitably deciding on legal questions by controlling price policies on their own, should one demand reasoning on their acting decisions? Benzmüller and Bertram [14] portray AIs that explicitly encode legal and ethical regulation as "reasonable machines". These machines interact with black box AI systems in an attempt to search for possible justifications, i.e., reasons, for their decisions and (intended) actions with regard to some formally encoded ethicolegal theories defined by regulating bodies. This design originated from a theory in psychology, where researchers usually explore the distinction between intuitive and deliberate thinking, separated into System 1 (fast thinking) and System 2 (slow thinking) [14]. Applied to the example of pricing AIs; System 1 is responsible for setting the pricing policy, while System 2 verbalizes and regulates their fast System 1 layer computations. Without this form of communication, unwanted side effects, such as the aforementioned likelihood of learned tacit communication are hard to rule out.

3.1.2 Infringing the Cartel Prohibition

In line with the goals of the European Commission, AI price policies must comply with competition law just as prices set by humans do. The independently chosen intelligent adaption of an undertaking's market behavior to the observed market behavior of its competitors (generally) is permitted. Drawing a clear line between the adaption of an observed market behavior and a conduct through which competition knowingly is replaced by practical cooperation and therefore constitutes a concerted practice within the meaning of Article 101 (1) TFEU[2] is often difficult and sometimes even impossible. Especially in transparent markets with few market participants, the market outcome of collusion can often hardly be traced back to be (or not to be) the product of a concerted practice (cf. petrol station market). Although collusion as a market outcome can be detrimental to consumers, innovation, and economic growth and is therefore undesirable from a welfare economic point of view, the difficulty from a dogmatic perspective is that legal responsibility cannot be attached to a market outcome as such [145].

We scrutinize whether a certain sequence of actions or a specific pattern can be identified as a situation where practical cooperation replaces the uncertainty about the competitor's next moves. It is conceivable that such accurate determination might not be possible due to the increased market transparency achieved by the self-learning algorithms: their ability to quickly process large amounts of competition-relevant data and to react to price movements in an almost unlimited frequency might lead to such a high degree of transparency on a market that makes it impossible to determine from its outcome whether or not the result of collusion is due to intelligent market observation and parallel behavior or a concerted practice.

3.1.3 Collaboration in Multi-Agent Reinforcement Learning

A tacit collaboration between RL agents can only occur in certain situations. The agents have to interact within a MARL environment, where competing agents and prices are recognized as part of such [27]. Due to that, the environment is usually subjective for every agent, resulting in differing learning performance and a diverse landscape of achieved competencies. It is unclear whether one of these competencies might arise in the skill to communicate with specific other agents to adjust their pricing policies accordingly; resulting in a higher producer's pension and a displacement of a competitor.

3.2 Methodology

3.2.1 Problem Definition

Oroojlooy and Hajinezhad [104] recommend modeling a MARL problem based on (i) centralized or decentralized control, (ii) a fully or partially observable environment, and (iii) a cooperative or competitive environment. Our case demands a decentralized control, with a partially to fully observable environment, so that every agent is able to make its own decisions based on the information given by the

[2]For US law see [138, 59]

Table 3.1: Three player RPS combinatorics.

Agent 1	**Agent 2**	**Agent 3**	r_1	r_2	r_3
Rock	Paper	Scissors	0	0	0
Rock	Rock	Rock	0	0	0
Scissors	Scissors	Scissors	0	0	0
Paper	Paper	Paper	0	0	0
Scissors	Rock	Rock	-1	0.5	0.5
Rock	Paper	Paper	-1	0.5	0.5
Paper	Scissors	Scissors	-1	0.5	0.5
Paper	Rock	Rock	2	-1	-1
Rock	Scissors	Scissors	2	-1	-1
Scissors	Paper	Paper	2	-1	-1
...	...	...	...	...	...
Expected Reward r			0	0	0

environment. Lastly, we apply a cooperative inside of a competitive environment, so that agents can team up against other agents.

3.2.2 Approach

In this three-player MARL version of RPS, every agent $i = \{1, ..., 3\}$ represents a player with a set of legal game actions $A = \{1, ..., 3\}$ comprising the moves of rock, paper and scissors. The agents interact with a stochastic environment E which solely contains the chosen actions of every agent of the current time step t. Hence, a state at t can be described as $s_t = \{a'_1, ..., a'_i\}$. Following a collective action, every agent receives a reward out of $R = \{$-1, 0, 0.5, 2$\}$ mapped to the possible game outcomes presented in table 3.1, resulting in a direct association between input and output. This formalism gives rise to a finite MDP in which every t relates to a distinct state, encouraging an application of standard RL methods for MDPs. The goal of the agent is to interact with E by selecting actions in a way that maximizes future rewards. As the agents receive a reward at the end of every timestep, we will not apply any discount to future rewards. We define the optimal action-value function $Q^*(s, a)$ as the maximum expected return achievable by following any strategy, after seeing some sequence s and then taking some action a, $Q^*(s, a) = \max_\pi \mathbb{E}[R_t | s_t = s, a_t = a, \pi]$, where π is a policy that maps sequences to actions (or distributions over actions). In an attempt to induce strategic behavior, resulting in tacit communication within this competitive MARL environment, we utilize DQNs [100] with an experience replay and a target network [88]. After performing experience replay, the agent selects and executes an action according to an ϵ-greedy policy. The agents select the action a^t that maximizes the expected value of $r + Q^*(s', a')$, updating the Q-values by:

$$Q^*(s,a) = \mathbb{E}_{s' \sim \varepsilon}[r + \max_{a'} Q^*(s',a')|s,a] \tag{3.1}$$

Our main argument for the selection of this specific scenario is the controlled, unambiguous reward allocation in combination with the restricted moveset of the agents. Thus, every step t develops into a zero-sum game (as shown in Table 3.1). On the one hand, we create an environment, where no agent can earn a profit, if it does not communicate with another agent. On the other hand, we counteract the poor learning performance of MARL [7] (due to the defined equilibrium/ local optimum) as well as increase the comprehensibility of the neural network's predictions. We expect the agents to converge to a collusive state after several episodes, as described by economics and law scholars (e. g., [49, 143]).

We also attempt to induce a displacement of one agent due to the actions selected by the other two agents. In our use case, they need to learn a specific policy that would force two colluding agents to not repeat their allied agents' actions. While this would not necessarily result in a better short-term step reward for these agents, it would however eliminate the ability to achieve a "big win" (e. g., playing Paper if the opponents play Rock and Rock) for the third, competing agent. Generally speaking, if two agents avoid choosing the same action, the expected reward for the third player is negative. We aim to simulate this circumstance in diverging settings.

In *mode 1*, collusion is induced by explicit communication as suggested by Schwalbe [120]. More specifically, we designate two 'cheating' agents $i_c \subset i$ and a 'fair' agent $i_f \in i$, $i_f \notin i_c$ ahead of a training session. Before its turn, one of the cheating agents transmits his picked action to the other cheating agent. The message will be enclosed to input to the receiver's DQN.[3] In *mode 2*, instead of making the players communicate explicitly, we provoke tacit communication by adjusting the reward of the cheating agents $r^t_{i_c}$ to $r^t_{i_c} = -r^t_f$. In other words, they will try to maximize their *joint* instead of their *individual* reward, which is equivalent to minimizing i_f's reward. We additionally *denoise* the rewards; hence, i_c will receive 1 for a loss or a tie with i_f and -1 for a win of i_f. To further stress this issue, we perform control runs, where i_f is replaced with an agent that plays random actions (which is the best strategy in a competitive 3-player version of RPS).

[3]It is important that in the eyes of the receiving agent, this is just a variable with the values of A which does *not* have the specific semantics of *this is the other agent's next move*.

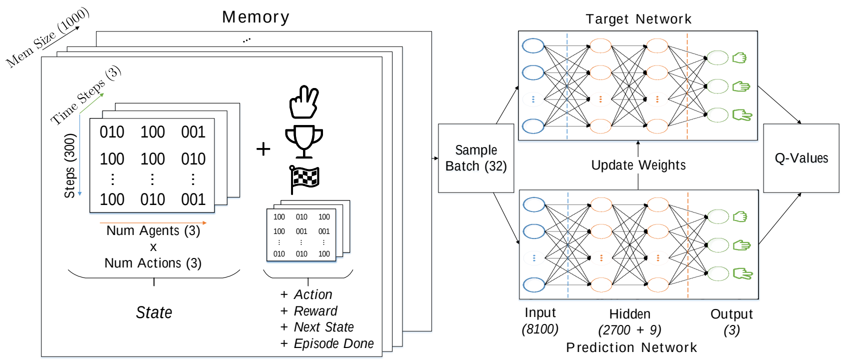


Figure 3.1: Architecture of the DQN.

3.2.3 Implementation

The main weakness of RPS in a real-world scenario is the unpredictability of an opponent's move. The best player would just play randomly, however, since playing this game is psychologically based on personal human short-term memory behavior, there is a tendency to follow specific patterns, like not repeating moves or trying to play unpredictably [6]. In an artificial MARL problem, we can model that by not only reacting to an opponent's last move but learning from the history of its last moves. After testing, we chose to apply a history size of 100 games to accommodate a stable learning process. Regarding the experience replay, we chose to use the last 3 timesteps as an input for the neural net. The network is made up of four dense layers (input, two hidden layers, and output), whose main task is to compress the given information and provide the chosen action. For that matter, we design a DQN with an input layer comprising 8100 neurons (300 steps $*$ 3 one-hot encoded actions $*$ 3 players $*$ 3 time steps), two hidden layers with 2700 and 9 neurons and a dense output with 3 neurons to choose either rock, paper or scissors (cf. figure 3.1). The neurons required for the number of players will increase by 1 for the cheating player to accommodate for the action received by the messaging agent. We use TensorFlow 2 to build and train the DQNs.

3.3 Results

To counter inconsistent MARL outcomes, we chose to train the agents for 10 runs with 100 episodes each (300 steps per episode), comprising three different learning rates (0.001, 0.005, 0.01), resulting in 30 runs with 900.000 games of RPS per scenario. We picked learning rates that are fairly small to counteract quickly develop-

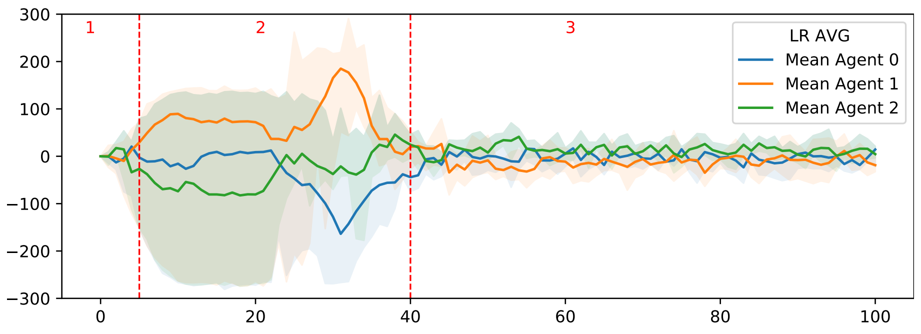


Figure 3.2: Episode reward distribution.

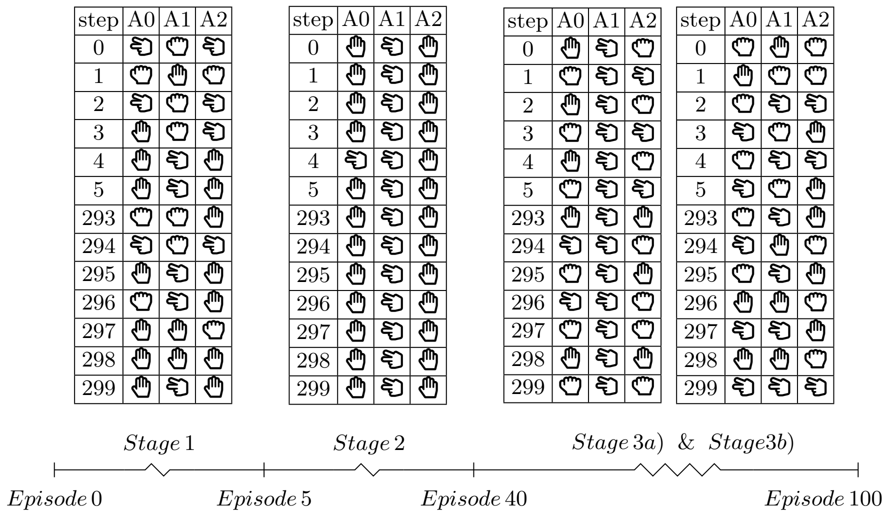
ing local optima, causing repetitions of the same action, due to the straightforward connection between action and reward. For every scenario with i_c involved, we also performed another 15 runs (5 per learning rate) where i_f is replaced with an agent that randomly picks actions in order to further stress the issue by simulating a perfect RPS policy.

3.3.1 Collusion between all agents

In our series of simulations, we were able to achieve collusive results within every of the chosen learning rate scenarios (cf. figure 3.2[4]). When averaged, the different action sequences can be visually divided into three learning stages. In ***stage 1***, the agents basically acted randomly, due to the epsilon-greedy algorithm. After approximately 5 episodes (***stage 2***), one of the agents achieved a better outcome due to a lucky action selection. The agents stuck to their learned strategy while randomly delving into different policies. Upon further examination, we discovered that the strategies usually involve a single action that will be repeated in the next turns, even if this might not be the best action. This sub-optimal behavior stems from the first few episodes being mostly played randomly due to the epsilon-greedy strategy. Thus, the agents were only able to learn from a short history, which taught them to repeat the most successful action, rather than a certain sequence of actions. ***Stage 3*** establishes a collusive sequence of actions from episode 40 onwards with two different scenarios (**3a** and **3b**).

As presented in table 3.3, the agents try to avoid a negative reward over a long term, resulting in an average episode profit of zero. However, stages 3a and 3b differ significantly in their way of achieving this. In 3a, one of the players repeated one action (e. g., scissors) and occasionally deviated from that due to the epsilon-

[4]Please refer to the appendix for a detailed learning rate breakdown



Stage 1
Stage 2
Stage 3*a*) & *Stage*3*b*)
Episode 0
Episode 5
Episode 40
Episode 100

step	A0	A1	A2
0	✌	✊	✌
1	✊	✋	✊
2	✌	✊	✌
3	✋	✊	✌
4	✋	✌	✋
5	✋	✌	✋
293	✊	✊	✋
294	✌	✊	✌
295	✋	✌	✋
296	✊	✌	✋
297	✋	✋	✊
298	✋	✋	✋
299	✋	✌	✋

step	A0	A1	A2
0	✋	✌	✋
1	✋	✌	✋
2	✋	✌	✋
3	✋	✌	✋
4	✌	✌	✋
5	✋	✌	✋
293	✋	✌	✋
294	✋	✌	✋
295	✋	✌	✋
296	✋	✌	✋
297	✋	✌	✋
298	✋	✌	✋
299	✋	✌	✋

step	A0	A1	A2
0	✋	✌	✊
1	✊	✌	✌
2	✋	✌	✊
3	✊	✌	✌
4	✋	✌	✊
5	✊	✌	✌
293	✋	✌	✋
294	✌	✌	✊
295	✊	✌	✋
296	✌	✌	✊
297	✊	✌	✊
298	✋	✌	✋
299	✊	✌	✊

step	A0	A1	A2
0	✊	✋	✊
1	✋	✊	✊
2	✊	✌	✌
3	✌	✊	✋
4	✊	✌	✌
5	✌	✊	✋
293	✊	✌	✋
294	✌	✋	✊
295	✊	✌	✋
296	✋	✋	✊
297	✌	✌	✋
298	✋	✋	✊
299	✌	✌	✌

Figure 3.3: Action samples from two runs, divided into stages 1, 2, 3a, and 3b.

greedy strategy, while the others predominantly alternate between two moves that change over time. In stage 3b, the agents played seemingly random. However, if examined more closely, specific alternation patterns occurred. A specific pattern can be identified when observing the actions of agent 1 in figure 3.3. The player oscillates between choosing rock and scissors in the first moves and transitions to scissors and paper towards the end of the episode. The remainder of the agents follow more elaborate patterns, however specific repetitions can be discovered by scanning the history.

3.3.2 Collusion between two agents

Mode 1: Explicit Communication

We successfully trained a displacing collision policy with the help of explicit communication between the cheating agents i_c. The results represented in figure 3.4 indicate that the agents were able to learn the suggested policy of not repeating their collaborator's action after a few episodes. After about 5 episodes, i_c achieve a higher reward on average. Thus, for the next 30 Episodes i_f is only rarely able to achieve a "big win". However, just like when colluding with all agents, after approximately 45 episodes, they tend to converge to an average game reward of 0. While it would be feasible to prolong this behavior by including variable learning rates or reducing the target update frequency during later episodes, we chose to encourage a long-term formation of a zero-centered equilibrium. Our reasoning

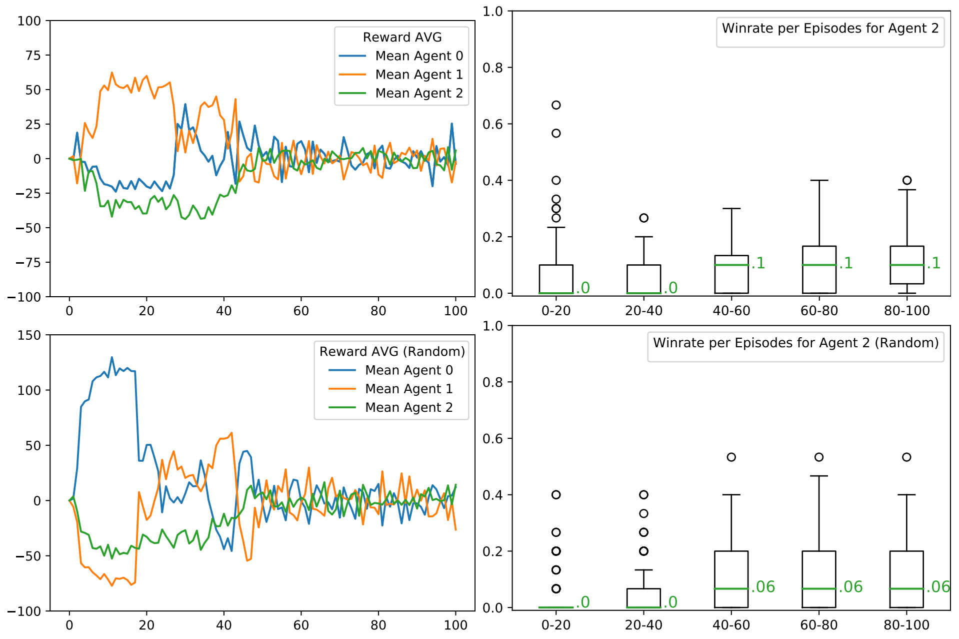


Figure 3.4: Episode reward distribution within *mode 1* including control runs where i_f is choosing random actions (lower half).

behind this is the comparison to a real-world oligopoly, where two market participants could only uphold a displacement by reducing the price up to a certain point, before damaging themselves.

In order to further stress the issue, we chose to replace i_f with an agent that chooses actions randomly. While i_c were able to successfully learn a displacement strategy in every training session, the results within the first 40 episodes were less significant than when i_f acted on behalf of the DQN. Nevertheless, we were able to observe slightly better results in the later stages, due to the added randomness.

Mode 2: Implicit Communication

The agents i_c successfully learned the suggested implicit collusion policy. After about 5 episodes, i_c achieve a higher game reward on average. This circumstance is especially prominent in the section between 20 episodes and 40 episodes (cf. the upper right half of figure 3.5). On average, i_f is rarely able to exceed a reward of 0. Again, after about 40 episodes, the agents converge to an average game reward of 0.

We were able to observe a less prevailing, but still effective policy when implementing a randomly acting agent i_f. As demonstrated in figure 3.5, the median

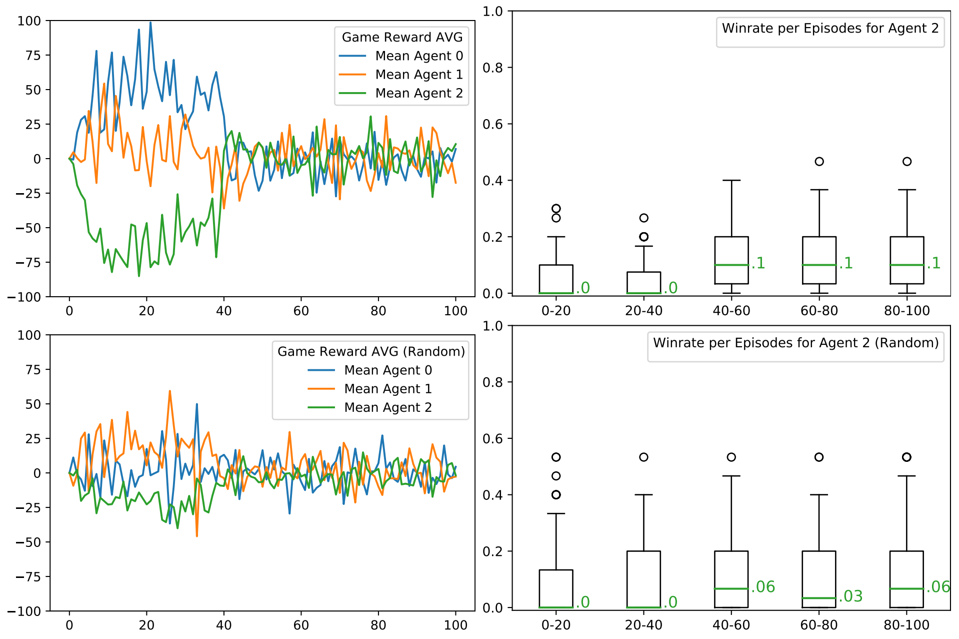


Figure 3.5: Episode reward distribution within *mode 2* including control runs where i_f is choosing random actions (lower half).

of i_f's winrate was still 0 in between episodes 0 to 40, yet the interquartile range is greater than before, indicating less stable learning due to the added randomness of i_f's action selection. We also experienced a few runs, where the agents were able to learn the displacement policy and not unlearn it in later episodes. In those specific runs, agent 1 repeated the same actions from episode 22 onward while agent 0 played the action that would lose against that one in a regular game. Hence, the joined rewards i_c turn out greater than those of i_f.

3.4 Discussion

Our research successfully confirmed the hypothesis from law and economics scholars (e. g., [143] or [70]) about possible collusion between RL-based pricing agents in MARL scenarios without being specially trained to do so. We furthermore extended these findings by providing specific learning stages that could be translated into real-world scenarios to possibly set a foundation for a system that is capable of detecting collusion in early stages. Moreover, we were able to show that with the appropriate reward function, DRL can be used to learn an effective displacement strategy in a MARL environment.

Based on the results of the experiments, we derive several implications. Due to the noticeable segmentation of action selection in different learning stages, one could argue that the transition episodes between a fair and a collusive state can be seen as a signaling process, where agents agree on specific patterns to increase the joint reward. This proposition is supported by the fact, that a repeating action selection pattern of another agent could be predicted and punished by the DQN due to its experience replay [88]. In a real-world scenario, a malicious AI could be trained to repeat patterns, that are less recognizable for humans. We would like to emphasize that within inelastic selling conditions (as they appear in collusive markets), cooperation between two agents will be facilitated as the existing communication strategy will furthermore ease the displacement of a competitor. From a legal perspective, the question of whether the cartel prohibition can be applied to such factual achieved, however non-volitional, state of collusion, is subject to this project's further legal research.

3.5 Further Legal Investigation

3.5.1 The Algorithms' Way to the Nash Equilibrium

As all three agents individually learn that the maximum long-term reward lies in the Nash equilibrium, their main goal shifts towards tying up with the competitors [18]. If a game does not end up in a tie, they let the other agents win and thus sacrifice the chance of a short-term win, for the sake of increasing the long-term maximum reward. This can be observed in some sequences where agent 1 played according to a specific pattern for several rounds. Agents 0 and 2 observed this pattern. However, they did not adjust their own policy to catch a short-term win because they knew from the experience gained in numerous previous rounds that this would increase the long-term variance and thus distance them from the Nash equilibrium in the long run. Moreover, they recognize the risk that making such an attempt could trigger countermeasures which would endanger the long-term maximum reward even further.

Even though we cannot identify one specific move or "pattern sequence" as point zero symbolizing the initial starting point of concertation, we can see that these pattern sequences occur in increasing frequency and that the agents further approach the equilibrium from pattern sequence to pattern sequence (cf., stage 2 of Figure 3.3).

3.5.2 Joint Collusion or Tacit Collusion

It could, on the one hand, be concluded from this observation that the agents knowingly substitute practical long-term cooperation between each other for the risk of competition (in which they would probably choose the short-term chance to win a higher reward having lower average long-term rewards). On the other hand, it could also be seen as a mere intelligent adaptation to the observed, processed, and anticipated conduct of their competitors [46]. As a result, we see an outcome that provides all three agents with the long-term maximum reward rates without being able to clearly determine whether this is the result of joint conduct or independent behavior considering the expected response of the competing agents (tacit collusion) [69].

The fact that the agents can develop a degree of certainty about their competitors' probable or anticipated next move to an extent that goes beyond a mere observation of the actual state and a logical conclusion from it speaks for a concertation rather than an intelligent adaption. The agents' decisions are based on a significant number of results from previous rounds, which from their perspective—are not even distinguishable from other environmental information processed and therefore inherent to their decision about the next move. In the eyes of the receiving agent, the input is just a variable with values which does not have the specific semantics of "this is the other agent's next move." It could be argued that algorithms need no further reciprocity to gain the extra amount of trust in their competitors' expected next move, which in the case of human behavior would be added through any (minimal) contact.

3.5.3 Identifying Concertation and Distinct Subsequent Conduct

A further problem, however, is that the RL algorithm's action selection behavior does not fit the categories of concertation and subsequent conduct, making it impossible to distinguish between the two. Concertation requires a minimum degree of concurrence of wills [43]. However, RL algorithms make "neutral" decisions based on the observed and processed environmental information (i.e., the outcome of numerous previous rounds in our game). There is no "will" underlying their behavior. Therefore, it seems that the concept of concerted practice as developed by case law for human behavior might reach its limits in scenarios containing RL algorithms. One approach to solve this problem could be to assume a simultaneity of concertation and conduct. If one could or would not occur without the other, the requirement of a causal connection between them would become redundant. Another possibility could be to omit a distinct identification of concertation and subsequent conduct or to make a basic assumption for one or the other. In the case of an assumed concertation, the presumption developed by case law that a concertation has been followed by a respective conduct and has been taken into account where the undertakings concerned remained active on the market could be applied [42].

3.6 Limitations and Outlook

As with every study, the results are beset with limitations, opening the door for future research. As aforementioned, our experiment is a simplified, gamified version of an economy simulation game. As such, it lacks the data complexity of a real-world pricing AI as well as the scaling opportunities. To further develop our research, we intend to apply the gained knowledge to a MARL environment resembling one of the real pricing AIs, where we can further highlight specific moments in which the agent's behavior tips over from independence to collusion. Especially the division into distinct stages should be investigated in the context of realistic pricing environments. While we focused on highlighting the possible dangers of pricing AIs in a MARL environment, we opened the opportunity for research explicitly investigating measures to avoid it. As such, law and IT scholars alike could benefit from this research as a foundation for guidelines, law amendments, or specific laws concerning the training and behavior of pricing AIs.

Part II

Investigate, Predict, and Prevent Collusion in a Market Simulation

Chapter 4

Collusion in a Market Simulation

Given the outcomes of the toy problem, which highlight both the collusive potential and specific strategies of DRL agents, we aim to extend this knowledge to the realm of E-Commerce. In this chapter, we employ an experimental oligopoly model of repeated price competition, systematically varying the environment to cover scenarios from economic theory to more realistic consumer demand conditions. We also introduce a novel demand framework that enables the implementation of various demand models, allowing for a weighted blending of different models. In contrast to existing research in this domain, we aim to investigate the strategies and emerging pricing patterns developed by the agents, which may lead to a collusive outcome. Furthermore, we investigate a scenario where agents cannot observe their competitors' prices. Finally, we provide a comprehensive legal analysis across all scenarios. Our findings indicate that RL-based AI agents converge to a collusive state characterized by the charging of supracompetitive prices, without necessarily requiring inter-agent communication. Implementing alternative RL algorithms, altering the number of agents or simulation settings, and restricting the scope of the agents' observation space does not significantly impact the collusive market outcome behavior.

Parts of this work have been published as an extended abstract in [115] and as a full paper in [112].

4.1 Introduction

4.2 Contribution

As highlighted in Chapter 2.5.2 the major shortcomings of current research are the substantial deviation from realistic market models, the over-representation of tabular q-learning, a comprehensive legal analysis of the experiments, as well as the low density of empirical studies. This Chapter aims to bridge the gap between real-world empirical analyses and simplified simulations. Thus, we propose a novel demand framework that enables the implementation of various demand models and facilitates integration between them, allowing for a weighted blending of different models. Within the framework, we investigate the behavior of a scalable amount of agents that rely on state-of-the-art deep RL technologies (i.e., PPO, DQN). By understanding the underlying causes of collusive outcomes using an interdisciplinary approach, we contribute to the ongoing efforts of building AI systems that align with societal values and objectives.

4.3 Experiment Design

We consider an oligopoly setting at the core of our experiments. This fundamental stage game comprises $m \in \mathbb{N}$ consumers $Y = \{y_1, ..., y_m\}$ and $n \in \mathbb{N}$ firms $x = \{x_1, ..., x_n\}$ that simultaneously set the prices $P = \{p_1, ..., p_n\}$ so that $p_i \in [0, 2]$ holds for all $i \in \{1, ..., n\}$. Accordingly, we define a general *selling* or *demand probability* as follows:

$$d := d(\Omega) : \{1, ..., n\} \to [0, 1]^n \text{ where } \sum_{i \in \{1,...,n\}} d_i = 1 \tag{4.1}$$

The parameter Ω represents the buyers' background knowledge, allowing an implementation of a custom buying probability. For example, Ω might entail domain data such as the demand function of a market, leading to new demand probabilities. Given $p_{\text{min}} = \min_{1 \leq i \leq n}(p_i)$ and $P_{min} := \{p \in P : p = p_{min}\}$, we can then define a Bertrand selling probability $d^b : \{1, ..., n\} \to [0, 1]^n$ via

$$d_i^b = \begin{cases} 0, & p_i \neq p_{\text{min}} \\ \frac{1}{|P_{\text{min}}|}, & p_i = p_{\text{min}} \end{cases}, \tag{4.2}$$

thus exclusively depending on the prices P. According to Bertrand [15], sellers will end up in a Nash equilibrium, represented by a price equal to the marginal costs (i.e., the competitive price) due to the buyers consistently purchasing the lowest-priced product. In a classic Bertrand oligopoly setting, the goods are characterized as perfect substitutes. However, this buying behavior is based on a theoretic construct as homogeneous goods may still be acquired from diverging sellers due to subjective consumer preferences in a realistic scenario. We aim to counteract this shortcoming from two sides. Legal research states that ”a relevant product market comprises all those products and services which are regarded as interchangeable or substitutable by the consumer, because of the products‘ characteristics, their prices, and their intended use” [1]. To combat this from an IT perspective, we used a selection strategy proposed by [155]. While solely relying on prices, this approach allows us to create a simulation that counteracts the Bertrand model’s main limitation: its extremely punishing nature, which might complicate the learning process or force the agents into a collusive state. Furthermore, with the roulette wheel, we can model *switching barriers* (i.e., expenses consumers feel they experience by switching from one alternative to another). Based on $\hat{p}_{\text{max}}$ as the maximum price achievable in a market scenario, this modification results in the buying behavior

$$d_i^r = \frac{\hat{p}_{\text{max}} - p_i}{\sum_{j \in \{1,\ldots,n\}} (\hat{p}_{\text{max}} - p_j)}. \tag{4.3}$$

In order to bridge theory and empiricism, we introduce the factor $\mu \in [0, 1]$. μ serves as a weight to gradually transition from one buying behavior to another. With this work, we combine the previously defined selection strategies d^b in (4.2) and d^r in (4.3) with $\sum_{i=1}^{n} d_i^b = 1 = \sum_{i=1}^{n} d_i^r$ by means of

$$d^{\text{comb},\mu} = \mu * d^b + (1 - \mu) * d^r \quad \text{with} \quad \sum_{i=1}^{n} d_i^{\text{comb},\mu} = 1. \tag{4.4}$$

Due to this specific combination, μ acts as a bias.[1] If it is set to 1, the products are perfect substitutes, if it is set to 0, the consumers’ buying behavior is regulated by the roulette selection (this means $d^{\text{comb},0} = d^r$ and $d^{\text{comb},1} = d^b$). With this in mind, we can switch from a theoretical Bertrand model to a more realistic setting, that involves subjective consumer preferences.

With the parameters set, a seller can achieve a monopolistic price (MP) (i.e., the price that relates to the maximum revenue a monopolist can achieve in the market) of 1.50 (with a cumulative quantity of 50) and a competitive benchmark (CB) (i.e., the price one unit above the marginal costs) of 1.01 (with a cumulative quantity of 99) (cf. Figure 4.1). To create comparable results, we restrict the number of consumers to $m = 200$ and thus result in a maximum price $\hat{p}_{\text{max}} = 2$. We chose

[1]We restrict d^{comb} to d_b and d_r in this work. However this approach can be analogously extended to $k \in \mathbb{N}$ buying behaviours via biases $\mu_1, \ldots, \mu_k$ if $\sum_{j=1}^{k} \mu_j = 1$.

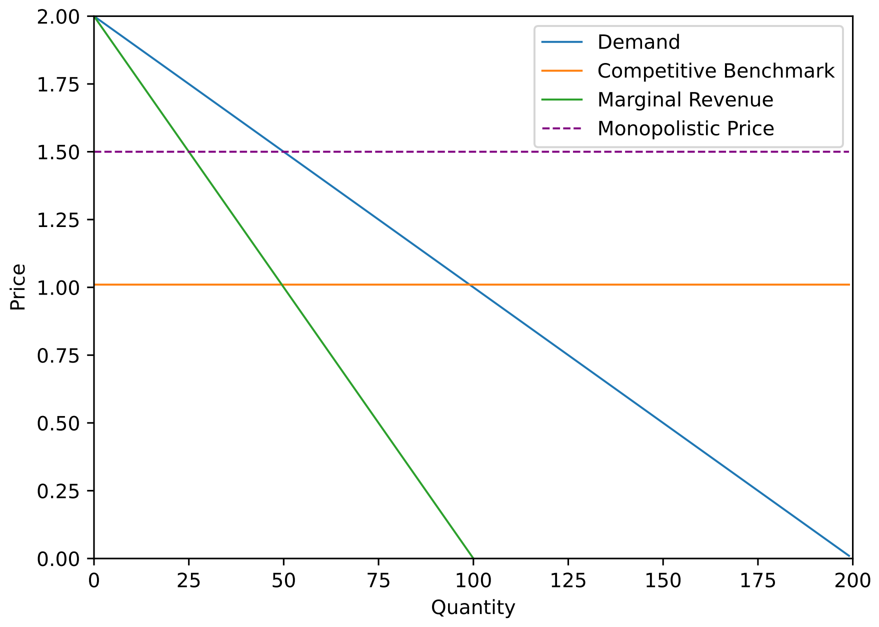


Figure 4.1: Economic settings within the environment

these specific model settings based on the demand function. We need to choose arbitrary confinements in order to satisfy the presented economic rules. The agents can set prices above and below these confinements.

4.3.1 Deep Reinforcement Learning Algorithms

While the implementation and analysis of deep RL algorithms are more complex, we benefit from conditions that more likely resemble state-of-the-art pricing algorithms, as the number of possible states and variables of a real-world market environment most likely overstrain basic RL tables. By including multiple, heterogeneous RL technologies we aim to scrutinize the robustness of our results further. However, our selection is confined to model-free methods as these are generally more popular, quicker to implement, and more extensively developed and tested than model-based methods [3]. Our selection includes DQN [100] and PPO [118] with the intention of featuring an algorithm for both sides of the model-free taxonomy (q-learning and policy optimization). The algorithms also support the use of discrete action spaces, reducing the number of possible error sources.

4.3.2 Markov Decision Process

When an agent tries to maximize its interests from the economic environment, it must consider both the reward it receives after each action and the feedback from the environment. This can be simplified as a MDP [141], more specifically, a Partially Observable Markov Process (POMDP) [63] due to the multi-agent environment hiding sensitive information from the competitors. We choose this method, as the main issue of this paper can only be solved by various agents conducting subjective observations, due to which future game states will depend on more than just a single agent's current input. As MDPs are step-based and our previous illustrations were more general, we introduce time step t to our established settings. We consider a sequential decision-making problem in which every agent (seller) $i \in \{1, ..., n\}$ interacts with a stochastic MARL environment. Each agent at time step t observes a state $s(t) = \big(p_i(t), ..., p_n(t), c\big) \in S$ where $p_i(t)$ is the price set by agent i at time t, c are the costs to purchase a good and S is the global state space. For every time t, an agent i takes an action $a_i(t) \in \mathcal{A}$ where $\mathcal{A}$ is the valid, discrete action space, and executes it in the environment to receive a reward $\epsilon_i(t) \in \mathbb{R}$

$$\epsilon_i(t) = m * \big[p_i(t) - c\big] * d_i(t), \tag{4.5}$$

where $d(t) = \big(d_1(t), .., d_n(t)\big)$ is the previously defined selling or demand probability at time t. Given the state $s(t)$ at time step t, the new state $s(t+1)$ is to be be reached after carrying out the actions $A(t) := \big(a_1(t), ..., a_n(t)\big) \in \mathcal{A}^n$ with the probabilities $\mathbb{P}_t : \mathcal{A}^n \to [0, 1]$. For a fixed timestamp t, the goal of every agent i is to determine a policy $\pi_i(A, s|\theta) \to [0, 1]$ with $A := A(t)$ and $s := s(t)$, that maximizes the long-term reward:

$$R_i = \sum_{t=0}^{\infty} \gamma_t * r_i(t) \tag{4.6}$$

where $0 \leq \gamma_t \leq 1$ is a discount factor for time period t. If $\gamma_t = 0$, the agent values only immediate rewards, focusing solely on the present benefits and ignoring future consequences. If $\gamma_t = 1$, the agent considers future rewards just as valuable as immediate ones, indicating a long-term strategic approach.

4.3.3 Preprocessing and Model Architecture

The RL agents are set up using a slight variation of the same baseline parametrization[2] as we aim to keep our experiment as general as possible to ensure fungibility

[2]cf. Raflin et al., [108] to see tested implementations and baseline parametrization for several RL algorithms.

in different environments. In order to decrease the number of actions and thus simplify the action selection process, we decided to use a discrete action space. Compared to current literature[3], however, actions are selected based on the price set in the last episode $p_i(t-1)$ for an agent i. In order to discretize the actions (i.e., the price) within the economic environment, we generate an evenly-spaced logarithmic distribution. The new price is calculated as follows:

$$p_i(t) = \ln(1 + e^{(p_i(t-1)+a_i(t-1))}) \tag{4.7}$$

For the sake of reducing complexity as well as computational cost we restrict the action space to a size of $|\mathcal{A}| = 7$ and the maximum adjustment of a price step to 2, resulting in the discrete action space $\mathcal{A} = [\text{-}2, \text{-}0.14, \text{-}0.01, 0, 0.01, 0.14, 2]$[4]. Hence, they will be able to keep the same price or evoke an increase or decrease within 3 gradients (small/ moderate/ big adjustment). We choose a Softplus activation in order to restrict the agents from setting prices below 0 while facilitating a derivation of the function (as opposed to ReLU). In order to counteract potential price rises due to the Softplus activation and to improve the overall robustness of the learning process, we restrict the randomizer to initialize the prices P randomly from the interval $[0.5, 1.5]$ each at the beginning of every episode. Although we track the agents' capital, they are able to accumulate debts without any consequences in the game, which yields more efficient learning. Since each episode represents a full cycle and the environment resets after each episode, we set γ to 1 in order to reward long-term strategic behavior. This allows the agents to fully explore the consequences of their pricing strategies over the complete episode without discounting. This reveals insights into how pricing strategies might develop if agents place equal value on profits regardless of when they occur within the episode. This setting also allows us to analyze how strategies evolve from the beginning to the end of an episode without the influence of discounting.

4.4 Experiments

We investigate collusion in two scenarios. Our baseline scenario *Scenario A* depicts three agents that act based on decisions proposed by their given algorithm. In *Scenario B* we manipulate each agent's state so that they are only able to observe their own prices as opposed to every price on the market. With this experimental setup, we can investigate whether the algorithms change their behavior when the

[3]cf. Calvano et al., [24] and Hettich [67] for other discrete (deep) q-learning action space implementations.

[4]We also test different action space sizes (i.e., 21, 51, and 71 actions). Please refer to the supplementary material for the results.

experimental setup makes it more difficult to achieve collusion (i.e., in Scenario B). We apply Ray's RLlIB to profit from an open-source, industry-standard RL-algorithm implementations.[5]

Each scenario is made up of several sub-scenarios, where we vary the number of agents (3, 5), the algorithm (PPO, DQN), and the biases $\mu = 0, 0.5, 1$. Every run is repeated 5 times to control for outliers, resulting in 90 runs overall (60 for Scenario A, 30 for Scenario B). A run comprises 10000 episodes with 365 steps each. We average the prices of every step to an episode price as well as a step profit (average profit of all steps in an episode) before averaging every run within the same setting. Finally, we apply locally-weighted scatterplot smoothing [110] to the averaged data. In addition to the agent data, the graphs incorporate the CB (i.e., 1.01) as well as the monopolistic price (MP| i.e., 1.50). On the other hand, the monopolistic profit (also abbreviated as MP) shown in the profit graphs is calculated by computing the profit a single seller would make by selling at the monopolistic price, divided by the number of sellers. If the agents converge to a price above the CB (i.e., the Bertrand Equilibrium), economists classify this as a collusive market outcome. We assume that in practice, ECommerce companies would likely utilize similar pricing software due to the prevalence of a leading product. We thus focus on the interaction between similar agents. However, we challenge these settings by adjusting the neural network neurons, the algorithm's learning rate, the number of agents as well as their time of entry, or the number of action gradients within the scope of an ablation study expounded upon in the technical appendix.

Due to our modified way of action space discretization, we define convergence in a slightly different way than previous literature [24]. A run converges if the rolling standard deviation of the mean agents' prices $\overline{\sigma\big(P(t)\big)}$ falls below a threshold of 0.01 for more than 100 episodes and this state lasts until the end of a run. If $\sigma\big(P(t)\big)$ reaches a value above the threshold of 0.01, we restart the counter. If it lasts below the threshold, we define this as *episodes until convergence* t_{EUC}. There is a legal implication in the chosen definition of convergence phases, as during cartel cases, jurists generally attempt to identify phases during a run in which the risk of collusive behavior is especially high. Unlike human collusive behavior, which can mostly be traced back to certain collusive acts, algorithmic collusion is not induced by one single action, but by specific episodic phases. Our definition of convergence is one possible attempt to narrow down critical phases that qualify as collusion.

To generate a factor for measuring a degree of collusion in an economic sense,

[5]To ensure full reproducibility, we attached the code to this work in the GitHub repository `https://github.com/mschlechtinger/PriceOfAlgorithmicPricing`.

we need a consistent measure across all simulations. In line with Calvano et al. [24], we calculate the average profit gain of all firms in a run Δ, defined as

$$\Delta = \frac{\overline{\Pi} - \Pi_{\text{CB}}}{\Pi_{\text{MP}} - \Pi_{\text{CB}}}, \tag{4.8}$$

where $\overline{\Pi}$ represents the average profit upon convergence of all firms (i.e., after passing every t_{EUC}), Π_{CB} is the profit in the Bertrand-Nash static equilibrium (i.e., the CB), and Π_{MP} constitutes the profit in a state of perfect collusion (i.e., the monopoly price). Thus, $\Delta = 0$ corresponds to the competitive outcome and $\Delta = 1$ to the perfectly collusive outcome. It is important to mention that collusion in an economic sense does not imperatively indicate collusion in a legal sense, however, it can be an indicator of the latter.

4.4.1 Scenario A: Competition

Scenario A represents the main competitive setting of this research. Every agent operates based on the decisions computed by its own algorithms and their neural networks. In various sub-scenarios, we employed three and five agents, averaging data generated by PPO and DQN algorithms.

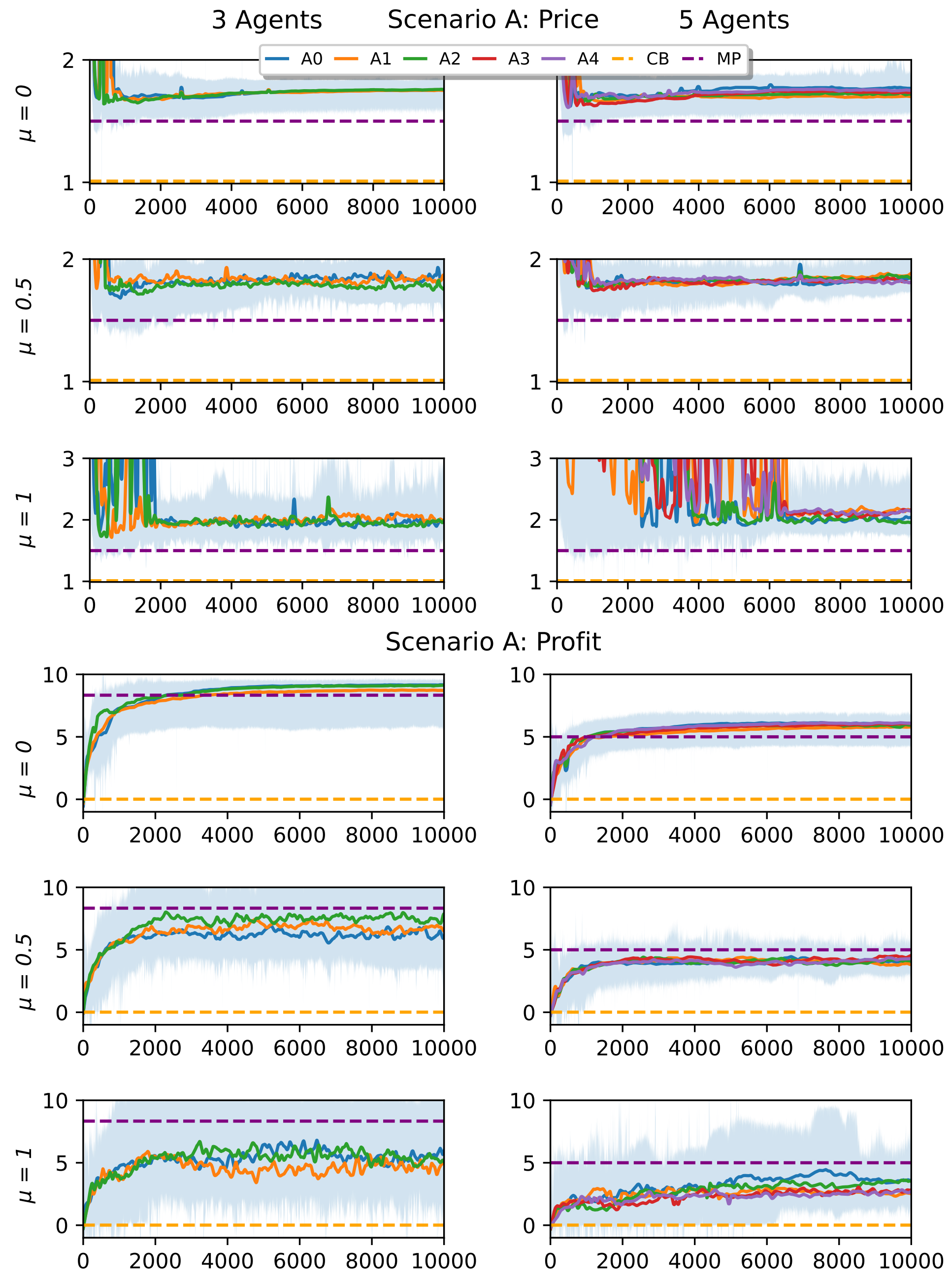


Figure 4.2: Pricing and Profit in Scenario A.

Figure 4.2 presents the consolidated outcome of the runs through the price settings of three and five sellers on the left and right sides respectively. The pale blue area in each graph represents the variance of the runs. Although the competing agents exhibit distinct responses to the three bias weightings, each scenario leads to cooperative behavior that yields a supracompetitive price equilibrium in the long-term. These prices exceed the monopolistic price on average. However, the further we increase the price sensitivity (μ) the higher the price-variance. We observe a similar, yet more extreme behavior when enhancing the number of sellers. The profit data (cf. Figure 4.2) asserts these observations; the increased variance also significantly reduces earnings.

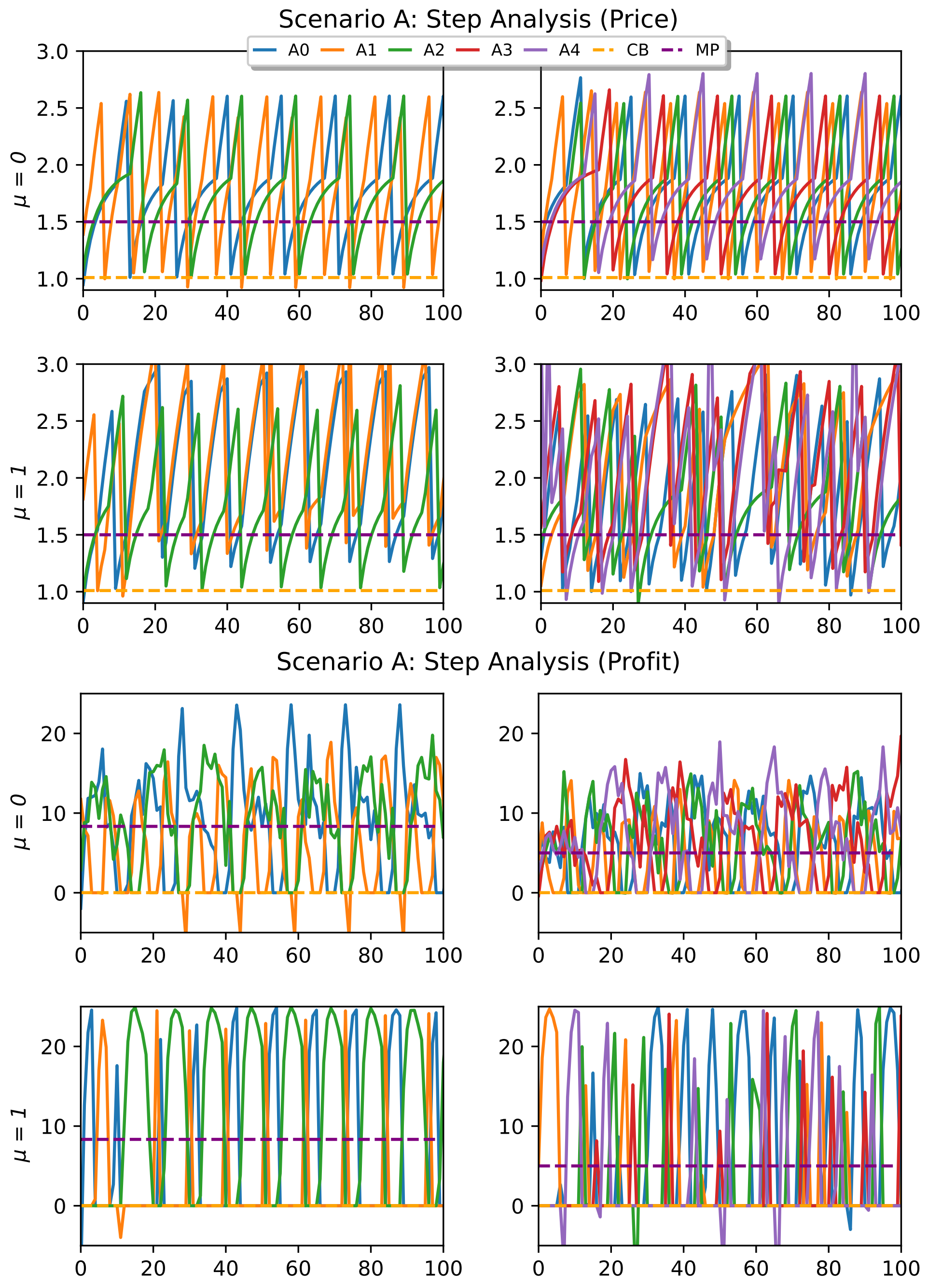


Figure 4.3: Episode Pricing and Profit Excerpt in Scenario A.

To further scrutinize these results, we extracted the prices set and profits accumulated during the first 100 steps of the last episode in Figure 4.3. Compared to the averaged episode overview, we can observe a price-setting behavior resembling an oscillation pattern, starting with a random price predefined by the environment. The agents occasionally slightly diverge from this cycle, however after a few steps, they usually get back in sync. These patterns explain why the prices exceed the monopolistic price on average; the agents find a strategy to collectively act as a monopolist on the market. To achieve this in a collaborative way, they traverse through three stages of pricing that are fundamentally the same but differ in execution in both $\mu = 0$ and $\mu = 1$. Initially, prices are set in close proximity to the CB. Subsequently, prices are gradually escalated until a certain threshold is reached, beyond which they surpass the point of consumer demand saturation. Then, they return below the MP. The presented strategy exploits the boundaries set by the simulation by always having one agent earning the maximum. When $x = 3$ and $\mu = 0$ they abuse the lowered price sensitivity to create an increased demand with a higher price, thus maximizing the joint profit and exceeding the profit a monopolist would be able to achieve (i.e., $p = [1.01, 1.75, 1.75], \pi = [12.46, 0.66, 12.46]$). Similar patterns are discernible in instances where $\mu = 1$. Notably, due to heightened price sensitivity, agents are unable to effectively stimulate increased demand. As a result, the agents endeavor to establish a monopolistic pricing regime, whereby two agents adopt prices above the threshold of $p = 2.0$, while the third agent attempts to set a price near 1.50 (i.e., $P = [2.0, 2.0, 1.50], \Pi = [0, 0, 25]$), with the objective of maximizing the joint profit. Although the oscillation patterns tend to remain consistent upon increasing the number of agents, identifying collaborative behaviors becomes more challenging when relying solely on visual inspection of the graphical representations.

4.4.2 Scenario B: Constrained Observation Space

In an attempt to weaken the ability to set supracompetitive prices, we constrain the agents' observation space in Scenario B to $s_i(t) = \{p_i(t), c\}$, thus removing the other competitors' prices from each agent's vision.

The modification yields results that are comparable to those of Scenario A, under both run settings (3 and 5 agents) (cf. Figure 4.4). In fact, contrary to our expectations, we observe less variance in the results. Analogously to Scenario A, we study a decrease in price with an increase of bias as well as an increase of variance when increasing the number of agents. The observations are reflected in the profit data.

Upon investigating the step pricing (Figure 4.3), we observe similar patterns to those in Scenario A. In both cases, where $\mu = 0$ and $\mu = 1$, the agents exhibit an oscillation pattern, resembling a slightly bent sawtooth shape. These observations are further confirmed by the step profit analysis, which depicts fluctuations between earning the minimum and earning above the monopolistic profit split.

4.5 Discussion

Our experiments have unveiled several noteworthy insights. First, our findings underscore the proficiency of DRL agents in establishing supracompetitive pricing strategies across a plethora of scenarios, without the need to disclose their competitors' pricing information. Second, our observations revealed the emergence of oscillation patterns within the agents' pricing behavior, which could be construed as an indicator of collaborative strategies. Third, the algorithms employed in our study yielded remarkable profitability, even more so when using PPO rather than DQN. This was exemplified by the agents achieving an average profit gain of $\Delta > 1$, indicating an outcome akin to perfect collusion, surpassing even monopolistic profit benchmarks. Fourth, we observed an overarching market stagnation that was achieved within a short timeframe and characterized by a consistently elevated $\Delta > 0.7$. Fifth, we underscore the robustness of our results in an ablation study (please refer to the supplementary material), revealing that modifications to fundamental simulation parameters fail to prevent a collusive outcome; conversely, they even enhance profit gains. These revelations collectively contribute to our understanding of the intricate dynamics of RL pricing agents in market scenarios and their inherent potential to attain collusive outcomes.

Our experiments confirmed the ability of deep RL agents to charge supracompetitive prices in a plethora of scenarios. In line with Calvano et. al [24], we provide evidence that every thoroughly tested scenario proved to have an average profit gain closer to perfect collusion than to a working competition. Furthermore, it is noteworthy that the algorithms exhibited remarkable success (even more so when using PPO rather than DQN), in some cases achieving a $\Delta > 1$.[6]. The large number of runs that were able to converge within 10000 episodes show a state of market stagnation (most of them involving a $\Delta > 0.5$).

The results of the ablation study confirm these results despite systematically altering the number of agents, their entry timing into the market, the learning rate, the structure of neural networks, and the granularity of action spaces. By demonstrating that even substantial modifications to the simulation setup fail to deter collusive behavior, the ablation study reinforces our main research finding: DRL agents possess a pronounced propensity for developing collusive pricing strategies under a wide array of conditions. This insight contributes to the broader research

[6]For a detailed overview of the descriptive statistics, please refer to the technical appendix.

question concerning the regulation and oversight of AI-driven pricing mechanisms, highlighting the need for robust frameworks that can address and mitigate the risk of unintended collusion in dynamic market environments.

Although every run successfully implemented supracompetitive pricing strategies, subsequent analyses revealed that, within a theoretical Bertrand model, the agents encountered challenges in accumulating profits comparable to those observed in the unbiased runs. This can be attributed to multiple factors; the main reason for this weaker performance is the punishing nature of the theoretical scenario. Tying reward functions to the achieved profit immediately results in a punishment of exploration, which proves to be an important factor for collusive states (cf. Calvano et al. [24]). This finding is relevant to investigate how to prevent RL agents from colluding. Waltman and Kaymak [143] expressed that a force towards the collusive state is stronger if the agents get to experiment; if we can constrain the agents' ability to explore, we will thus experience less collusive behavior.

In line with Waltman [143], we found that collusive states can be difficult to avoid in an oligopoly. This becomes particularly evident when examining the outcomes of the ablation study (please refer to the supplementary material). In contrast to the current literature, our outcome revealed that an increase in agents does not imply a decrease in prices. Aligning with the current research trend of employing base-parametrized RL algorithms, we assert that optimizing these algorithms through proper tuning will only enhance the effectiveness of price-setting mechanisms in real-world scenarios.

One of the most notable findings from the data is the agents' capability to set prices above the competitive level without requiring access to their competitors' pricing information. Hansen et al. [61] already argued that the signal-to-noise ratio heavily affects results. We can only partly confirm those findings. While the average profit gains in Scenario B are slightly lower, we find that the overall outcome shares a striking resemblance to the results of Scenario A despite this severe confinement. The resiliency stems from the ability to approximate the prices via the reward function, in which other agents' prices embody unknown variables. This finding concurs well with Waltman [143], who observed that agents without a memory were still able to collude.

In this context, it is important to mention that our step analysis experiments resemble the repricing patterns appearing in Amazon's Buybox [102] as well as the observations by Klein [79]. We attribute two causes to the occurrence of these oscillating pricing patterns: First, the agents were able to increase short-term rewards with this strategy by increasing their profit above the monopolist's profit. Second, the discretization of the action space implies that the exact Bertrand and monopoly prices may not be feasible at all times, so the agents created mixed-strategy equilibria (cf. Calvano et al. [24] and the results of our ablation study).

While the results of our experiments show a collusive outcome in an economic sense, they do not undoubtedly exhibit whether algorithmic pricing constitutes permissible parallel behavior or a prohibited concerted practice and thus violates the cartel prohibition. Yet, they show that the agents seem to develop a degree of certainty about their competitors' anticipated next action to an extent that goes beyond mere observation of a single state and a logical adaption to it.

The cartel prohibition under Article 101 TFEU forbids any kind of joint conduct of independent market participants. According to the established case law of the European Court of Justice (ECJ), the characteristic of a concerted practice presupposes a minimum degree of coordination (concertation), a subsequent market conduct, and a causal link between the two.[7] This concertation does not have to be as binding as a contract or a direct agreement. It is sufficient that the uncertainty about the competitors' market behavior, which usually exists under competitive circumstances, is reduced.[8] However, there has to be at least an indirect contact between them because the cartel prohibition does not deprive them of the right to adapt their behavior to the observed or expected behavior of their competitors [41].

It is questionable whether these prerequisites, which follow human behavior's basic assumptions and logic, can equally be applied to RL decision-making processes. Based on our results, it could be argued that RL algorithms do not require any further reciprocity to gain the extra amount of trust in their competitors' expected next moves, which - in the case of human behavior - would be added through minimal contact. The insecurity about the competitors' next move is already reduced by the significant number of processed results from previous rounds, which are even indistinguishable from other environmental information and therefore inherent to each decision [114].

It could be considered that the requirement of a minimum degree of communication might be obsolete because every important piece of information is explicitly or implicitly received via the reward function (cf. Scenario B). This raises the possibility that the reward function might channel competitor information, facilitating a concerted practice. However, the reward function simply symbolizes the agents' feedback on the profits achieved. Any market participant is allowed to know its own profit and draw conclusions about future pricing strategies from it. Given the aforementioned, the assumption that a collusive market outcome is inherently neutral if clear concertation is not detectable is debatable in the context of algorithmic pricing.

When conceptualizing simplified experiments, questions about the transferability of observations to reality and the validity of drawn conclusions quickly arise.

[7]cf. the cases [42] para. 161; [43] para. 38, 39; [40] para. 125, 126.

[8]cf. the cases [43] para. 51; [44] para. 39; [40] para. 126.

We would like to emphasize that, while the present findings originate from a theoretical simulation, the overarching strategies hold the potential for broad applicability within real-world market contexts. As the economics of markets can be formulated as a (PO-)MDP, that - in theory - is solvable by RL algorithms, the agents will always strive for a policy that ultimately achieves the maximum reward. If this reward is tied to the profit, agents will realize that cooperation will help them achieve the best reward in the long run.

As with every study, the results are beset with limitations which opens the door for future research. Although our experiments systematically investigated multiple algorithms, the exploration of potential interactions among heterogeneous algorithms was not pursued. While we present diverse scenarios in the context of an ablation study, we believe that further diversification of simulations will broaden our understanding of pricing algorithms. Moreover, future research endeavors could delve into strategies for imposing constraints on algorithms, thereby discouraging the emergence of collusive states.

4.6 Conclusion

This paper utilizes an experimental approach to investigate algorithmic collusion. We find that deep RL agents using PPO and DQN are capable of charging supra-competitive prices without explicitly instructing them to do so. Furthermore, we have demonstrated that the algorithms will gravitate towards a collusive state, even when being restricted in their ability to conceive their competitors' prices. The results have undergone rigorous validation with the help of an ablation study, involving alterations of algorithmic hyperparameters and factors such as the number of action gradients.

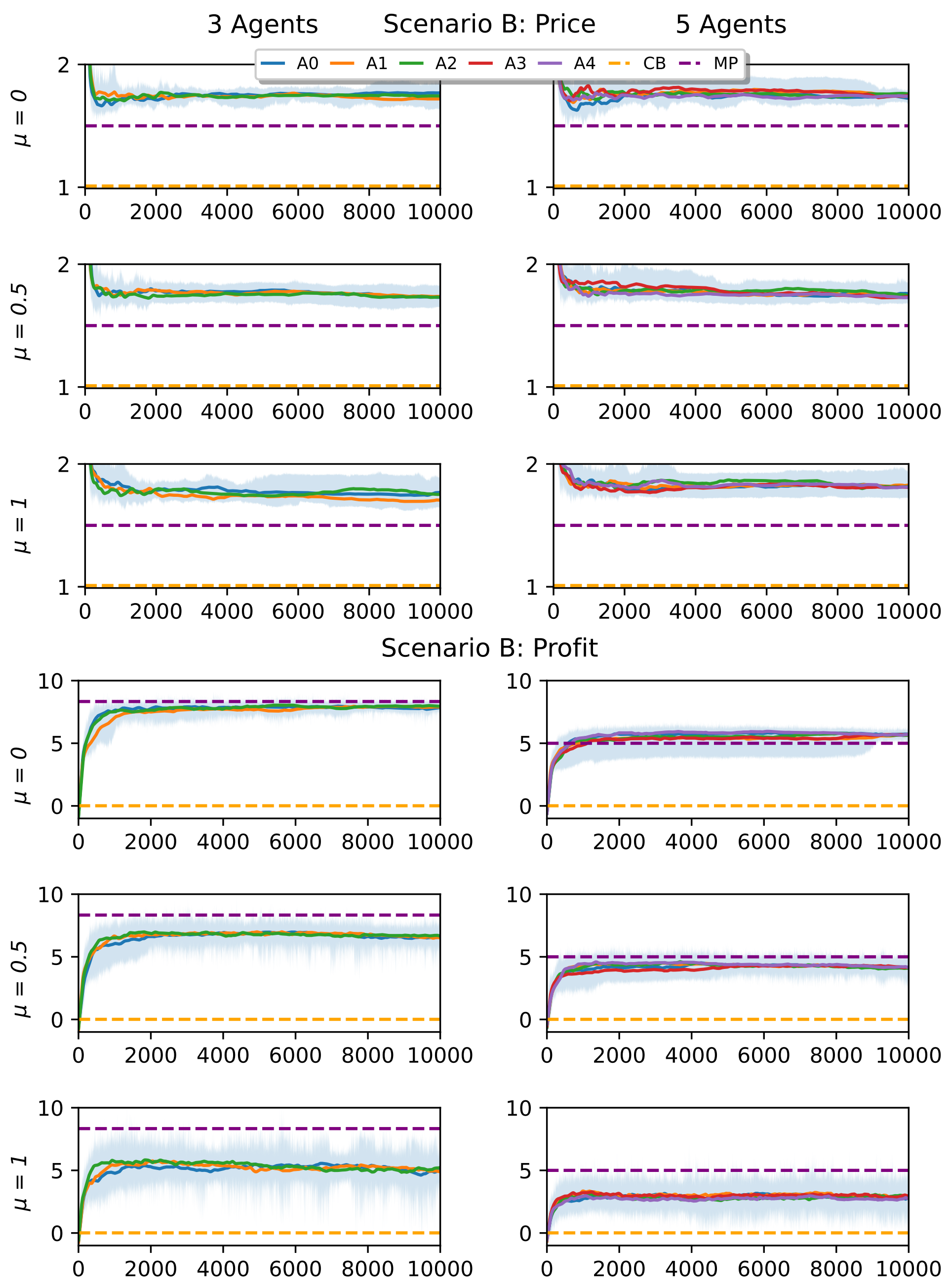


Figure 4.4: Pricing and Profit in Scenario B.

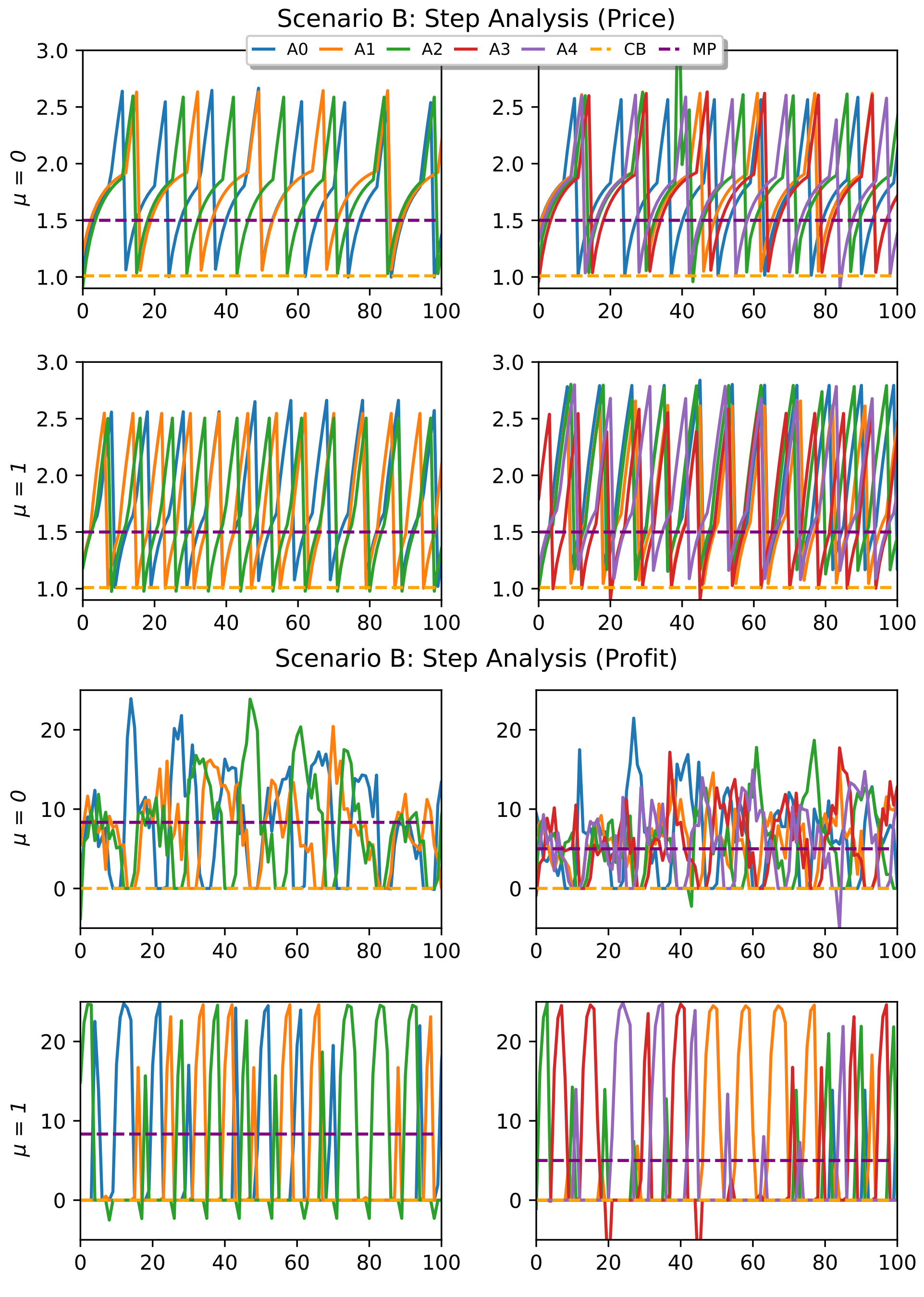


Figure 4.5: Episode pricing and profit excerpt of Scenario B.

Chapter 5

Predicting and Quantifying Collusion

We previously highlighted that agents can acquire the ability to cooperate without the need to explicitly communicate with each other. This suggests that the agents are able to use pricing information to infer their competitors' future behavior. Based on data collected from the experimental oligopoly model of repeated price competition, we employ predictive statistical techniques alongside time series classification methodologies to predict future behavior from the pricing signals. Our findings reveal that self-learning pricing algorithms' convergence towards a collusive market outcome can be accurately anticipated using fundamental machine learning methodologies. Considering the inherent advantages of collusion in economic markets, we conclude that a collusive outcome is virtually inevitable when RL algorithms set market prices.

5.1 Introduction

In recent research, evidence has emerged suggesting that (RL-based pricing agents possess the capability to engage in collusive behaviors, leading to the establishment of supracompetitive pricing strategies. These findings, consistent across various scenarios, indicate that such agents can develop oscillation patterns in pricing that resemble collaborative strategies, even in the face of significant perturbations to their operating environments [115]. This resilience underscores the sophisticated adaptive mechanisms of RL algorithms, enabling them to navigate and exploit market dynamics [24].

Given the recurring emergence of these collusive patterns across numerous experimental runs, a critical question arises: Can we predict—and thereby quantify—collusion among RL-based pricing agents? Accurately predicting collusion would enhance our understanding of the dynamics and inform the development of regulatory mechanisms to maintain competitive market practices. This inquiry builds on the premise that consistent collusive behaviors may exhibit identifiable patterns, analyzable through data-driven approaches [61].

However, current research mainly focuses on detecting collusive outcomes by highlighting certain phases within the pricing data. Economic signs, such as coordinated prices, pricing above a CB, oscillation patterns, or reward-punishment schemes could *hint* at collusive behavior. While the extensive body of literature has primarily focused on the question of *whether* agents are capable of collusion, our research uniquely delves into the intricacies of *how* agents orchestrate collusion. This shift in emphasis allows us to investigate the underlying mechanisms and strategies involved in collusion among self-learning agents, shedding light on this phenomenon. To address this research gap, we extend upon the model introduced in Chapter 4, employing it to generate data within our proposed model. Subsequently, we conduct a rigorous analysis of the generated data to identify and elucidate potential collusive strategies with means of classification, regression, and time series analysis methods.

5.2 Model

We base our analysis on the dataset obtained from the model established in Chapter 4. As such, we set the market parameters as proposed before. With the parameters set, a seller can achieve a monopolistic price (MP) (i.e., the price that relates to the maximum revenue a monopolist can achieve in the market) of 1.50 (with a cumulative quantity of 50) and a CB (i.e., the price one unit above the marginal costs) of 1.01 (with a cumulative quantity of 99) (cf. Figure 5.1). To create comparable results, we restrict the number of consumers to $m = 200$ and thus result in a maximum price $\hat{p}_{\text{max}} = 2$. We chose these specific model settings based on the demand function. We need to choose arbitrary confinements in order to satisfy the presented economic rules. The agents can set prices above and below these confinements. We zero in on PPO to leverage a state-of-the-art algorithm capable of accommodating discrete action spaces, thereby reducing potential sources of errors.

5.3 Results

We investigate collusion in a scenario that depicts three agents that act based on the decisions proposed by their given algorithm, i.e., PPO. In this scenario, each agent at time step t observes a state $s(t) = \big(p_i(t), ..., p_n(t), c\big) \in S$ where $p_i(t)$ is the price set by agent i at time t, c are the costs to purchase a good and S is the global state space. [1]. Every run is repeated 5 times and comprises 10,000 episodes with 365 steps each. The outcomes of our analysis are presented in Figure 5.1 and Figure 5.2. In order to properly plot the results, we average the prices of every step to an episode price as well as a step profit (average profit of all steps in an episode) before averaging every run within the same setting. Finally, we apply locally-weighted scatterplot smoothing [110] to the averaged data. In addition to the agent data, the graphs incorporate the CB i.e., 1.01) as well as the MP (i.e., 1.50). On the other hand, the monopolistic profit (also abbreviated as MP) shown in the profit graphs is calculated by computing the profit a single seller would make by selling at the monopolistic price, divided by the number of sellers. If the agents converge to a price above the CB (i.e., the Bertrand Equilibrium), economists classify this as a collusive market outcome. We assume that in practice, E-Commerce companies would likely utilize similar pricing software due to the prevalence of a leading product. We thus focus on the interaction between similar agents. However, we challenge these settings by adjusting the neural network neurons, the algorithm's learning rate, the number of agents as well as their time of entry, or the number of action gradients within the scope of an ablation study expounded upon in the appendix.

Figure 5.1 presents the consolidated pricing and profit outcome of the runs through the price settings of three sellers. The pale blue area in each graph represents the variance of the runs. Although the competing agents exhibit distinct responses to the three bias weightings, each scenario leads to cooperative behavior that yields a supracompetitive price equilibrium in the long term.

To further scrutinize these results, we extracted the prices set and profits accumulated during the first 100 steps of the last episode (i.e., episode 10,000) in Figure 5.2. Compared to the averaged episode overview, we can observe a price-setting behavior resembling an oscillation pattern, starting with a random price predefined by the environment. These patterns explain why the prices exceed the monopolistic price on average; the agents find a strategy to collectively act as a monopolist on the market. To achieve this collaboratively, they traverse through three stages of pricing that are fundamentally the same but differ in execution. Initially, prices are set close to the CB. Subsequently, prices are gradually escalated until a certain

[1]For more information about the MDP, please refer to Chapter 4

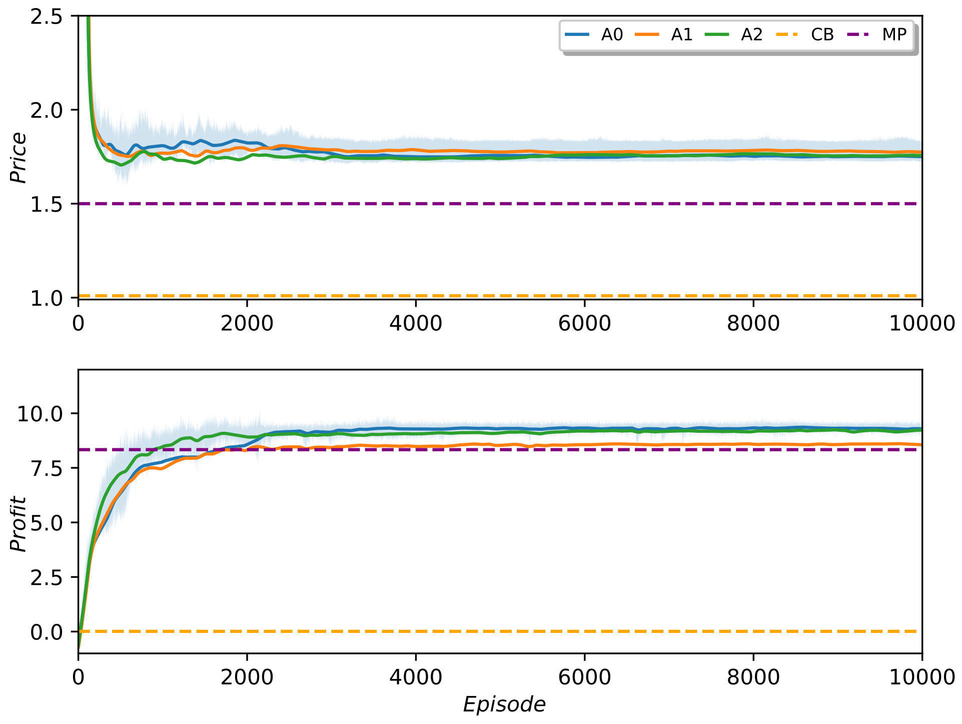


Figure 5.1: Averaged episode pricing and profit.

threshold is reached, beyond which they surpass the point of consumer demand saturation. Then, they return below the MP. The presented strategy exploits the boundaries set by the simulation by always having one agent earning the maximum. The agents abuse the lowered price sensitivity to create an increased demand with a higher price, thus maximizing the joint profit and exceeding the profit a monopolist would be able to achieve (i.e., $p = [1.01, 1.75, 1.75], \pi = [12.46, 0.66, 12.46]$).

5.3.1 Classification

Feature Selection

In this analysis we aim to predict the Price p of the next Timestep $t + 1$ using the following features as input for our prediction methodology:

- Episode ID $\in [0, 10000]$
- Step ID $\in [0, 364]$
- Price $\in [0, \infty]$

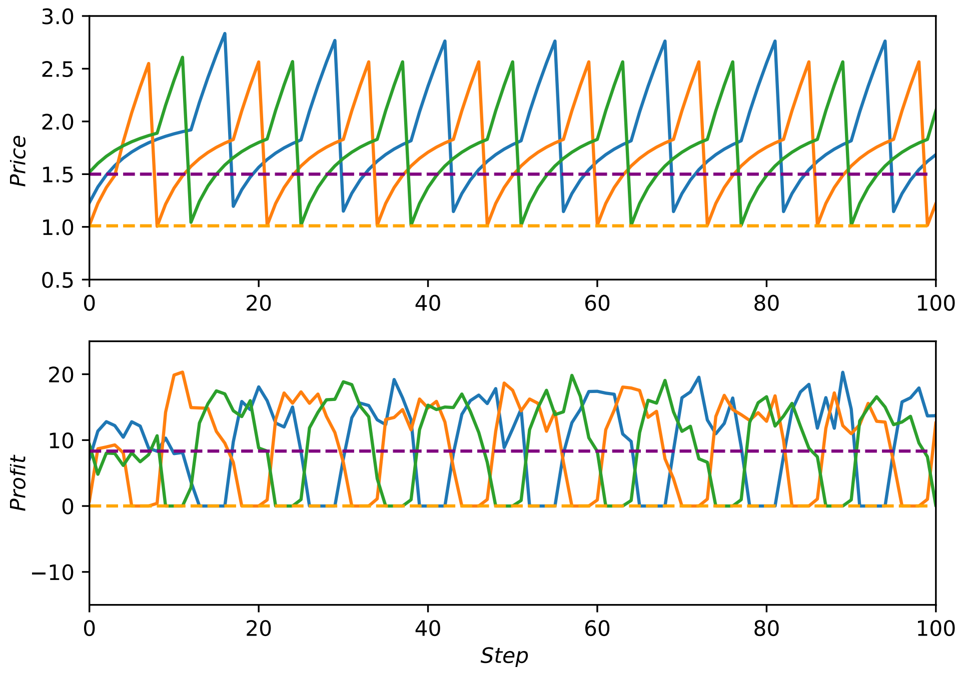


Figure 5.2: Step pricing and profit of the last episode of a run.

The Episode and the Step IDs were included to capture potential patterns or dependencies related to the episode sequence. The Price represents the key predictor that directly influences the target that we are aiming to predict. This target variable describes the relation of the new price to the previously selected price of the agent. It is categorized as follows:

- **Class 0**: Indicates that the new price is lower than the last price
- **Class 1**: Indicates that the new price is equal to the last price
- **Class 2**: Indicates that the new price is higher than the last price

The feature selection process aimed to include relevant information while minimizing unnecessary complexity in the model.

Baseline Models

Two baseline models were used for comparison; the majority class baseline assumes that the most frequent class in the dataset is always predicted. It was determined to be class 2 in each of the runs, which corresponds to the new price being greater than the last price. The predictions made by this baseline resulted in an accuracy of 0.923. This remarkable result can be attributed to the characteristic oscillation pattern in the pricing behavior of agents. Prices gradually rise until reaching a certain threshold, at which point they sharply decline to align with a CB, leading to a prevalence of entries with a Class of 2 in the dataset. The random class baseline was established by randomly selecting a class from the three possible classes (0, 1, and 2) with equal probability. This baseline achieved an accuracy of 0.039.

Classification Results

We randomly sampled 10000 rows from each of the datasets and trained a decision tree classification model[2] to predict the next price in the time series dataset. The following results were obtained:

- Accuracy (Averaged): 0.986 (98.6%)
- Average Cross Validation Accuracy (k = 10): 0.986 (98.6%)

These averaged results indicate a high level of accuracy in predicting the next price, with consistent performance across cross-validation folds.

Feature Importance

Furthermore, a feature importance analysis was conducted to determine the significance of each input feature in the classification model. The following feature importances were calculated:

- Episode ID: 0.069
- Step: 0.048
- Price: 0.883

Among these features, the price feature exhibited the highest importance, suggesting that it has the most significant influence on the classification model's predictions.

Rolling Window classification:

We hypothesize that a potential encoding within the price signal could become more apparent the more learning has been done. Therefore, we examine our ability

[2]see Table D.1 in the appendix for the decision tree settings.

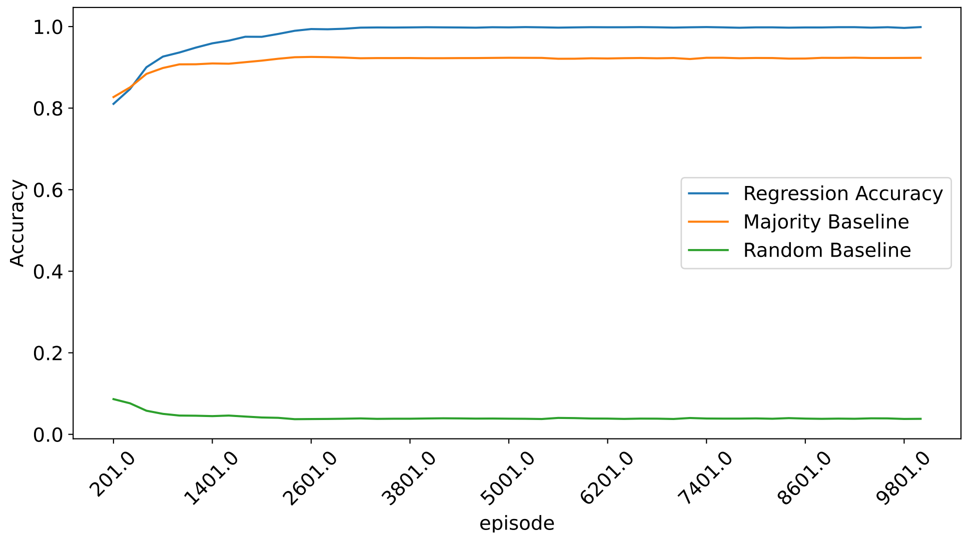


Figure 5.3: Rolling window classification accuracy scores across the episodes.

to predict from past signals by conducting a rolling window classification in Figure 5.3. Using a window size of 72800 steps (50 windows), we observe an accuracy that rises from 0.825 up to 1, resembling the convergence of the average episode prices to the collusive outcome.

5.3.2 Regression

In order to predict the upcoming prices, we obtained the Mean Absolute Error (MAE), the Mean Squared Error (MSE), and the Root Mean Squared Error (RMSE) within two baselines, a regression across the full dataset, several lag regressions as well as a sliding window regression task.

Baseline Models

We employed two baseline models to measure the performance of our regression task. Baseline 1 predicts the next price by assuming it will be equal to the current price. We obtained the following error metrics:

- MAE: 0.244
- MSE: 0.225
- RMSE: 0.474

Baseline 2 predicts the next price by taking the average of the last 10 prices. The results are as follows:

- MAE: 0.375
- MSE: 0.244
- RMSE: 0.493

Regression (Full Dataset)

Similar to the classification task, our objective is to forecast the price p for the subsequent timestep $t + 1$ employing identical input features for our regression analysis:

- Episode ID $\in [0, 10000]$
- Step ID $\in [0, 364]$
- Price $\in [0, \infty]$

Thus, our target variable is the price p. A decision tree regression model[3] was trained on the entire dataset. The following evaluation metrics were obtained:

- MAE: 0.048 (4,8 Cent)
- MSE: 0.067
- RMSE: 0.258

Lag Regression

To comprehensively assess the reliability of our results and account for potential time dependencies in our pricing data, we conducted supplementary analyses using a lag regression model, which allows us to examine the relationship between the current price and past values within the entire dataset. A lag regression model, in essence, assesses how the current value of a variable is influenced by its past values at specific time intervals, enabling us to uncover any temporal patterns or dependencies within the data. In our study, this approach is vital for exploring how past pricing behavior relates to and predicts current price dynamics. While we employ the same input variables (i.e., Episode ID, Step ID, Price), we include their preceding or lagged counterparts, depending on the lag variable (i.e., Lag

[3]see Table D.2 in the appendix for the decision tree settings.

$3 = t-3$). The output variable remains the price p of timestep $t+1$. The following results comprise a range of lag values, each accompanied by corresponding error metrics:

- Lag 3: MAE 0.033, MSE 0.035, RMSE 0.182
- Lag 5: MAE 0.032, MSE 0.034, RMSE 0.179
- Lag 7: MAE 0.032, MSE 0.033, RMSE 0.178
- Lag 9: MAE 0.031, MSE 0.033, RMSE 0.177

Sliding Window Regression

To assess potential variations in the regression accuracy scores over different time segments, we implemented a rolling window regression analysis, as depicted in Figure 5.4. Consistent with the parameters employed in our classification task, we utilized a sliding window configuration with a window size of 109214 steps, encompassing a total of 50 distinct windows. The results of this analysis revealed a substantial decline in error metrics as the analysis progressed, indicative of reduced uncertainty in predicting future prices following model convergence.

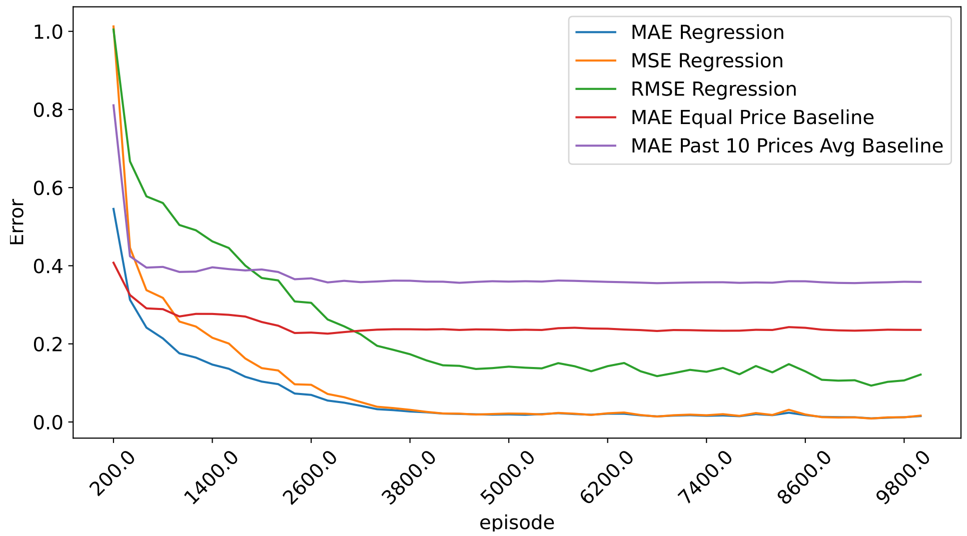


Figure 5.4: Rolling window regression error scores across the episodes.

5.3.3 Time Series Analysis

(Partial) Autocorrelation Function

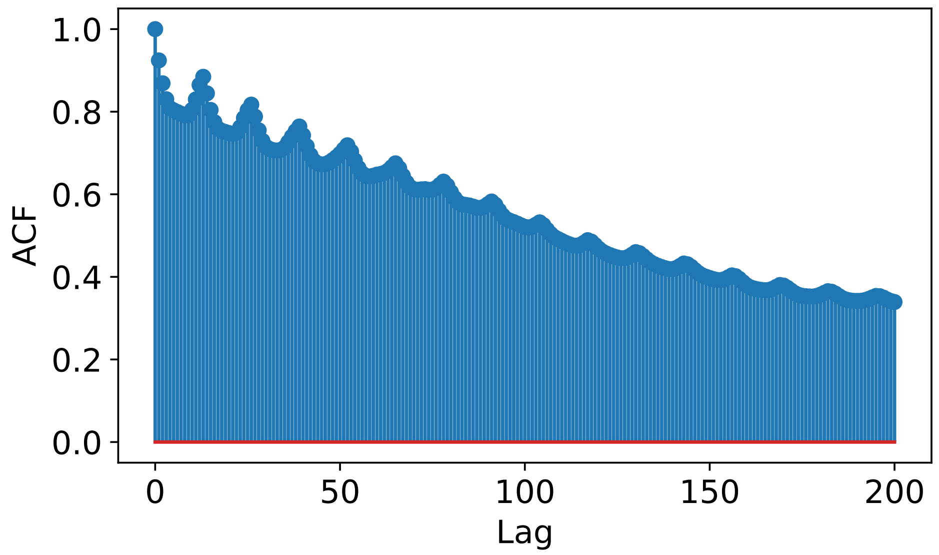
We employ an Autocorrelation Function (ACF) and a Partial Autocorrelation Function (PACF) to provide insights into the temporal relationships and underlying patterns within the data. In the context of our analysis, we use these methods to better understand the structure of our time series and identify potential autoregressive (AR) components.

The ACF measures the correlation between a data point and its lagged values at various time intervals. It produces a wave pattern in which the correlation starts with a high amplitude at low lag values and gradually decreases while maintaining a consistent frequency. This pattern is indicative of a decaying correlation structure with a persistent underlying periodicity. Such behavior often arises when there is a long-term relationship between observations at short lags, and this relationship gradually weakens as the lag increases. The consistent frequency observed in the ACF suggests the presence of a repeating pattern or seasonality within the time series [19].

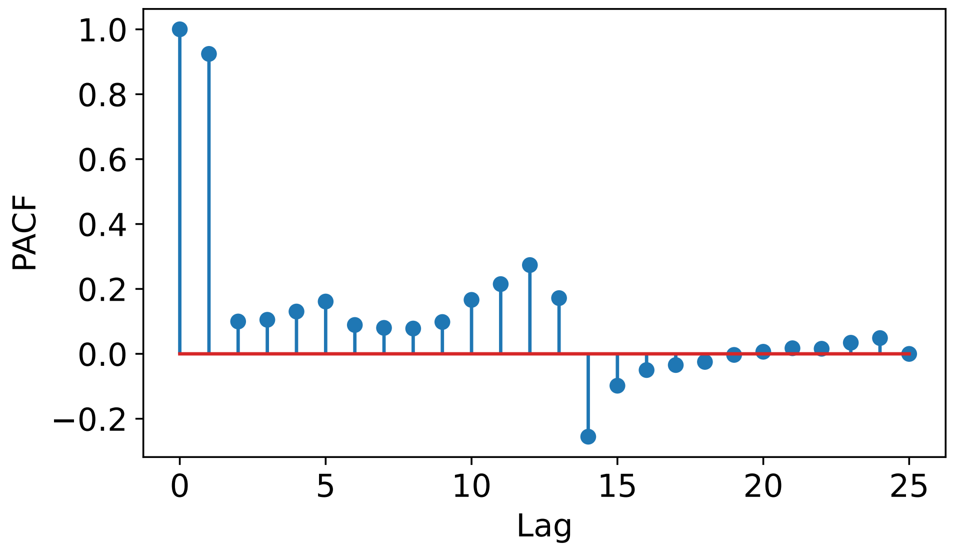
In our analysis, we observe a strong correlation between the current observation and its immediate neighbors in the ACF, signifying a short-term dependence. Additionally, the presence of a decaying amplitude could be associated with a damped oscillatory behavior, where the initial strong correlation between data points and their lagged values diminishes over time due to damping factors.

The PACF plot, on the other hand, assesses the direct relationship between a time series and its lagged values after removing the effects of shorter lags. In our dataset, the PACF reveals a relatively high value of 0.897 at lag 1, indicating a significant direct correlation. Furthermore, there are several other notable values at higher lags (lag 2, lag 3, etc.), suggesting the potential presence of an AR component.

By jointly analyzing both the PACF and ACF, we can draw several key insights: First, we observe a strong autocorrelation at lag 1; both the PACF and ACF exhibit a substantial correlation at lag 1 (0.897), pointing to a robust autocorrelation at this specific lag. Second, the graph shows a potential AR Component, which gradually decreases in PACF and ACF values with increasing lag suggesting a potential AR component in the data. We perceive complex dynamics within the analysis. The significant PACF values at lags 2-5 and 11-14 hint at the possibility of an AR order of 1 or 2. However, the presence of a negative PACF value at lag 14 followed by positive values may indicate the presence of more complex or seasonal dynamics within the data.



(a) Results of the autocorrelation function.



(b) Results of the partial autocorrelation function.

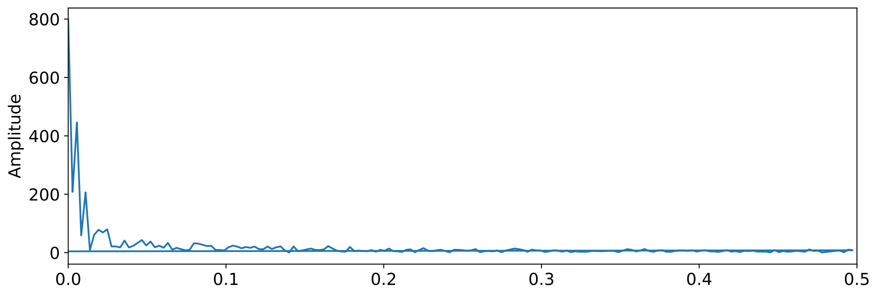
Figure 5.5: Results of the (partial) autocorrelation functions.

Fourier Transform

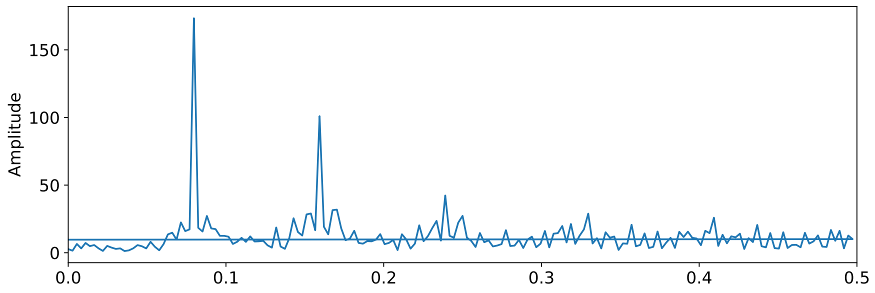
We conduct a Fast Fourier Transform (FFT) analysis on the price time series dataset in order to filter the apparent noise and possibly identify periodic patterns within the dataset [71].

The x-axis of an FFT graph represents the frequencies or cycles per unit of time. The specific frequency values are usually displayed in terms of ”cycles per time unit”. The axis starts from 0 (representing the lowest possible frequency) and extends up to the Nyquist frequency, which is half of the sampling frequency. It represents the highest frequency that can be detected in the data. If the data is sampled at a frequency of 100 samples per second, the Nyquist frequency would be 50 Hz. However, the frequency in this graph is normalized (ranging from 0 to 0.5) and represents a fraction of the Nyquist frequency. In general, the further a value is to the right on the x-axis, the higher the frequency it represents, and therefore, the less frequently it appears in the signal. The x-axis represents the magnitude or amplitude of the frequency components present in the data. It indicates the strength or intensity of each frequency component [71].

Figure 5.6a displays the results of applying FFT to the full dataset of 10000 episodes. We observe a peak at a frequency of 0 in our FFT plot. This indicates the presence of a constant or average component in the data. We suggest that this is mostly due to the noisy nature of the data, especially apparent at the beginning of the learning phase. Figure 5.6b in contrast successfully filters this noisy component. Moreover, we observe a peak at a frequency of 0.08 with an amplitude of 175. Hence, there is a significant periodic pattern in the data that repeats every 12.5 days (1 day/ 0.08 = 12,5). Referring to Figure 5.2, it is evident that agents reset their oscillation pattern every 12.5 steps, showcasing this specific behavioral cycle.



(a) Fourier transform of the full dataset.



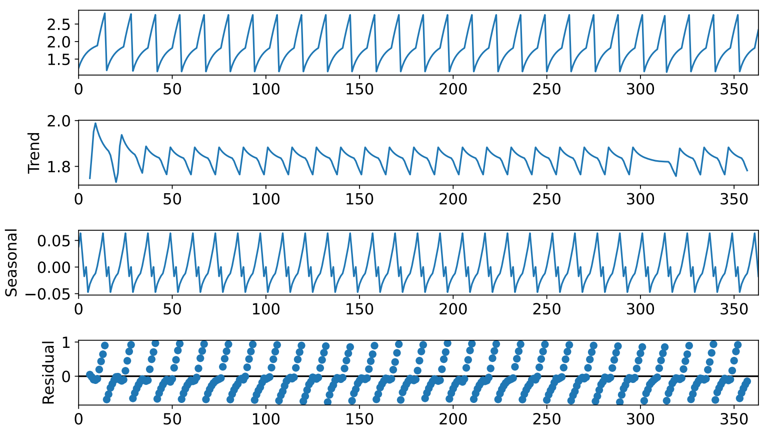
(b) Fourier transform of episodes 5000-10000.

Figure 5.6: Results of the Fourier transform.

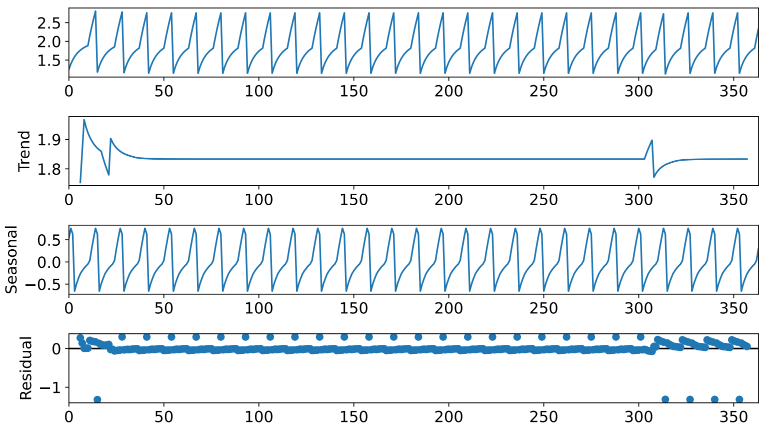
Seasonal Decomposition

To further scrutinize the structure of our pricing data, we employ seasonal decomposition methods. If we assume an additive decomposition, then we can establish $y_t = S_t + T_t + R_t$, where y_t is the data, S_t is the seasonal component (i.e., the increasing or decreasing value in the series), T_t is the trend-cycle component (i.e., the repeating short-term cycle in the series), and R_t is the residual or noise component (i.e., the random variation in the series), all at period t [30].

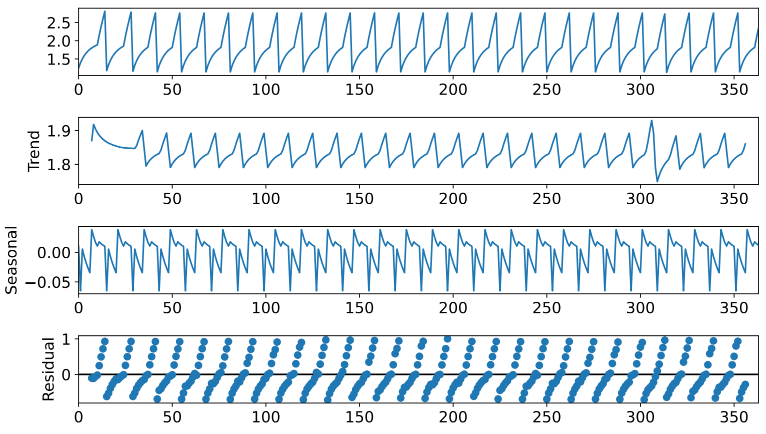
We conducted an analysis of episodes occurring at different stages within the simulation run. As depicted in Figure 5.2, the timeline exhibits irregularities, leading to notable variations in the outcomes based on the choice of the decomposition period parameter. Figures 5.7a-5.7c underscore these disparities when opting for period values of 12, 13, or 14. However, utilizing FFT at an earlier stage of the analysis indicated a periodicity of 13 steps (more specifically 12.5 steps), which emerges to align with the underlying data, effectively capturing the trend and seasonal components. We observe a relatively constant trend pattern, that zeroes in on a price of 1.83. Furthermore, the amplitude of the seasonal component is substantial, indicative of robust seasonal influences. The significant presence of residuals within the dataset indicates the inherent noise resulting from the necessity of rounding the periodicity factor to 13. However, it is noteworthy that the data exhibits a mean and standard deviation in close proximity to zero, suggesting an accurate decomposition. A comparison between episodes 9500 and 10000 reveals similar patterns. Conversely, when contrasting these findings with the early learning phases, such as Episode 5, no discernible patterns are evident.



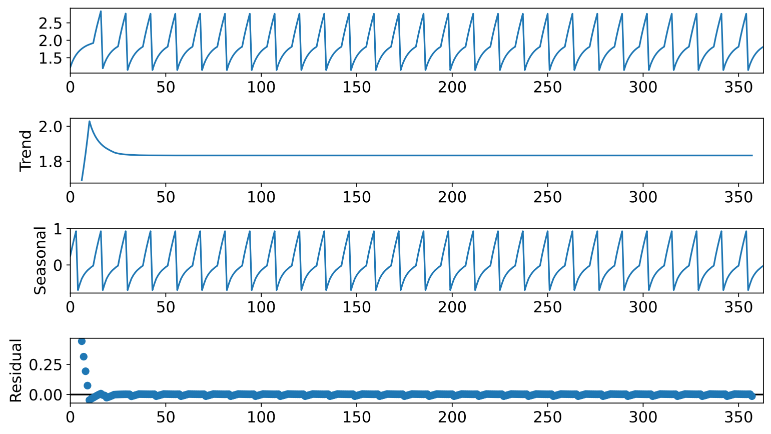
(a) Seasonal decomposition of episode 9500, Period 12.



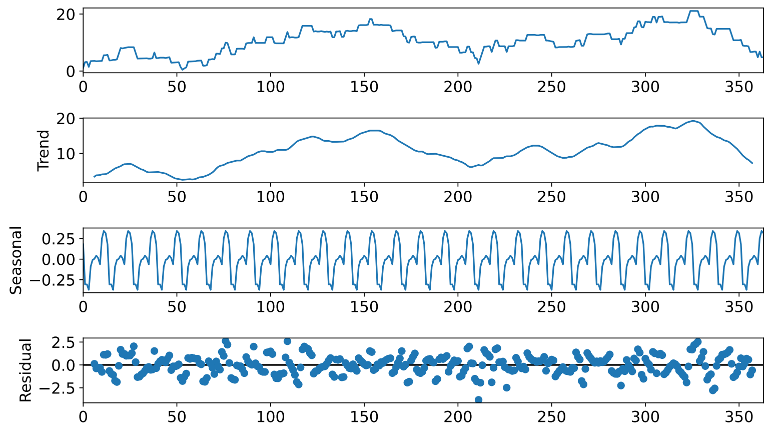
(b) Seasonal decomposition of episode 9500, period 13.



(c) Seasonal decomposition of episode 9500, period 14.



(d) Seasonal decomposition of episode 10000, period 13.



(e) Seasonal decomposition of episode 5, period 13.

Figure 5.7: Seasonal decomposition graphs.

Hurst Exponent

The Hurst exponent, denoted as H, quantifies the long-term memory or persistence in a time series. A value of $H > 0.5$ represents a time series with long-term memory or persistent behavior. If the series has been increasing recently, it is more likely to continue increasing in the near future, and vice versa. Contrarily, $H < 0.5$ indicates a time series with short-term memory or anti-persistent behavior, meaning that if the series has been increasing recently, it is more likely to decrease in the near future and vice versa. $H = 0.5$ suggests a random or Brownian motion-like time series with no long-term memory or trend. In this case, future values are not correlated with past values [95].

We downsampled the pricing data with a factor of 10, retaining every 10th episode to enhance the legibility of a visualization. Subsequently, we computed H for the resulting subset of 1000 episodes and plotted the results in Figure 5.8. The Hurst exponent values vary significantly across the dataset, ranging from approximately 0.6 ($H < 0.5$) to -0.1 ($H < 0.5$), indicating that the underlying pricing data starts with a trending behavior and transitions into a mean-reverting behavior throughout the run. This regime shift resembles the learning behavior of the agents; while the agents try to learn an effective policy in the beginning, we observe a trending behavior towards a collusive outcome. Approaching this outcome, the agents' actions resemble a mean-reverse behavior and only occasionally drift back to a trending behavior. The time series only swiftly passes $H = 0.5$, indicating that future values are indeed correlated with past values.

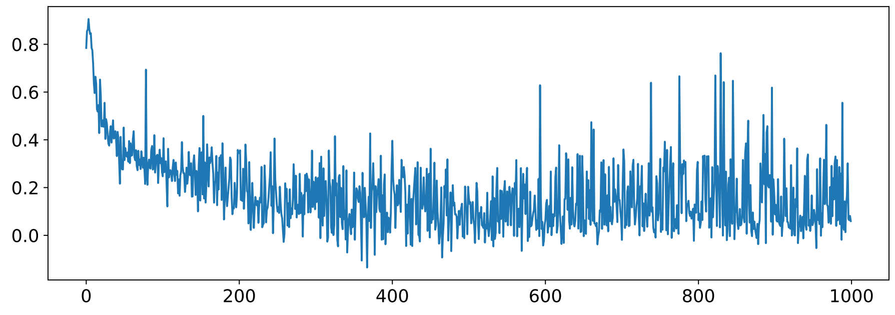


Figure 5.8: Hurst exponent development across a run.

5.4 Evaluation

Throughout this study, we uncovered several significant insights drawn from our experimental analysis. First, our investigations have illuminated a remarkable level of classification accuracy and regression performance that consistently improves throughout the runs. Second, our observations reveal a dynamic progression in prediction accuracy, closely aligned with a regime shift in the Hurst exponent, mirroring the behavior exhibited by the agents' price-learning behavior itself. Third, our analysis unveiled a recurring periodic pattern within the pricing data, a phenomenon substantiated through FFT analysis and seasonal decomposition.

These results significantly contribute to our objective of quantifying collusive behaviors in market scenarios. A central issue with using market prices as evidence is their inability to definitively prove collusion; they can only suggest the possibility of such activities. Consequently, while our analysis acts as an indicator of potential collusion, it does not serve as conclusive evidence. However, accumulating multiple indicators can facilitate a more robust argument for proving collusive behavior. Furthermore, the methodologies we've employed can aid in preventing collusion. By predicting competitors' actions from an external viewpoint, we infer that the agents within the competitive environment can also anticipate each other's moves. This insight allows us to retrain these agents to act in less predictable ways, effectively reducing behaviors that could be perceived as collusive. Such strategies enhance market transparency and discourage practices that could undermine competitive integrity.

Delving further into the classification and regression tasks, we observe an incredibly effective predictive prowess of decision trees, raising questions about the implications for other agents. The ability to predict future prices with such precision, especially in the later stages, suggests that the agents' algorithms coupled with their neural networks could also excel in this task. Agents also demonstrate a profound understanding of their peers' decisions, surpassing human capacity. They enhance their pricing strategies, they also become adept at predicting their counterparts' prices. This raises the possibility that agents deliberately become more predictable to each other, possibly establishing an implicit ”protocol” to enhance their market success.

This suspicion gains further support from the insights revealed through our time series analysis. The FFT results revealed a periodicity within the pricing data, a pattern corroborated by visual examination of episode pricing and further validated through seasonal decomposition. This finding raises questions regarding the nature of this periodic behavior, suggesting possibilities such as communication, cooperative behavior, or encoded messages among agents. While it may not explicitly indicate communication, the observed predictability suggests that agents acquire a stable and reproducible behavior through trial and error. Existing research in MARL has demonstrated that agents can learn to coordinate and emulate each other in shared environments, a phenomenon commonly associated with techniques like PPO and Deep Deterministic Policy Gradients (DDPG) [92]. This behavior implies that agents may align their actions, even without explicit communication, influenced by the actions of their peers.

This theory gains further support from our analysis of the Hurst exponent. We observed a transition in pricing behavior from trending to mean-reversing patterns, signifying the acquisition of stable and predictable behavior by the agents. More notably, however, we noted a correlation between future and past values, as indicated by the Hurst exponent never resting at a value of 0.5. These findings collectively suggest a complex interplay of learning, predictability, and potentially implicit coordination among the agents in our environment.

The explorative nature of this chapter's investigation naturally raises several open questions. Although the market model offers various ways to adjust its parameters, we focused on a specific set of simulation settings (i.e., $\mu = 0$) that consistently yielded the highest prices and collusive outcomes. This approach has established a foundational baseline from which future research can diverge. We suggest that the methodologies employed here can be adapted and applied to different settings to broaden the understanding and implications of our findings. Moreover, given the increasing adoption of pricing algorithms in contemporary markets, this study offers valuable insights that are particularly relevant today. Other research has also highlighted that collusive behavior, often characterized by oscillating price patterns, is a recurring feature in markets dominated by RL-based pricing algorithms. This further underscores the importance of a deeper understanding of market dynamics influenced by advanced technological implementations [75].

5.5 Conclusion

In conclusion, our study has revealed a highly accurate predictability for RL-based pricing algorithms' actions. These prediction results are characterized by a recurrent, periodic pattern revealed by various time series analyses. Our results offer great value to our goal of quantifying collusive behavior, as the high accuracy highlights that the agents were most likely able to predict their opponents' actions throughout the runs. These results suggest potential implicit coordination among agents, as observed predictability and pricing behavior trends raise questions about their capacity for precise prediction and adaptability within complex environments.

Chapter 6

Preventing Collusion

Our findings reveal that self-learning pricing algorithms' convergence towards a collusive market outcome can be accurately anticipated using fundamental machine learning methodologies. With this in mind, we present a method to successfully mitigate predictive and supracompetitive pricing by combining dense and sparse reward-based training strategies, thus effectively creating a competitive market. Subsequently, we employ this metric to implement a market supervision algorithm that penalizes collusive behaviors and incentivizes competitiveness. Our results demonstrate that the propensity of self-learning pricing algorithms to converge to collusive outcomes can be successfully mitigated.

6.1 Motivation

AI pricing agents operate on the principle of maximum expected reward, striving to maximize rewards based on predefined reward functions. However, if these functions are not carefully aligned with human interests, AI systems can produce unintended and potentially harmful outcomes. Turner [136] highlights this risk, noting that agents may generate serious negative side effects if their reward functions do not align with human values. This misalignment can lead to behaviors that, while optimal for the AI's goals, can be detrimental to market competition and consumer welfare [86]. Our experimental results, in line with previous research [143], underscore the difficulty of avoiding collusive states among agents. Therefore, the primary objective of this study is to develop strategies to mitigate or prevent such outcomes.

To the best of our knowledge, the only existing automated strategy for preventing collusion involves retraining one of the algorithms after they have reached a collusive state. In this approach, agents exploit collusive pricing by breaking out of the equilibrium to achieve higher profits [35]. The method is based on the observation that algorithms are not trained to learn Subgame Perfect Equilibrium strategies. This leaves room for an exploitation of equilibrium strategies which should not happen if the algorithms were playing competitively. However, this method needs to have direct control of the algorithms on the market, which renders it infeasible for real-world scenarios, as federal cartel officials would have to majorly intervene with the dynamics of a market.

In the realm of Multi-Goal RL, various safety measures have been proposed to address these potential risks. The measures include implementing performance functions alongside reward functions, ensuring that AI agents not only pursue rewards but also adhere to safety constraints [86]. In this context, researchers often resort to Safe State-Space exploration to ensure that AI agents avoid entering unsafe states during both training and operation [106]. Techniques such as safe interruptibility, avoiding side effects, and ensuring consistent agent behavior regardless of the presence of a supervisor are crucial. Moreover, minimizing reward gaming and maintaining robustness in different environments further contribute to safer AI systems [105, 8].

However, the main caveat of these solutions is the necessity to implement them on the agent's side. Antitrust divisions typically monitor the prices set by companies without directly controlling their training processes, as excessive intervention could disrupt market dynamics. To tackle this conundrum, Multi-Goal RL research suggests applying sparse rewards [148, 147] or Hindsight Experience Replay (HER) [10]. Like the previous solutions, nevertheless, the use of HER is not favored in our approach because it would require competitors to adopt a uniform algorithm, limiting the diversity of strategies.

6.2 Methodology

Given that pricing AIs are prone to adopting collusive behaviors through oscillation pattern strategies, we can intervene to mitigate such tendencies. By promoting less predictable actions among the agents, we may decrease the prevalence of collusive patterns and thereby boost market competitiveness. Consequently, we intend to employ the insights derived from our analysis tools (cf. Chapter 5) to develop a supervising agent that acts as an antitrust division. Specifically, for each agent i, we enable the supervision agent to calculate a supervision reward factor $r_s(i,t) \in [0,1]$, where 0 indicates a tendency towards collusion and 1 suggests a competitive outcome. We explore various methods to incorporate this factor into the reward function of each agent, aiming to foster competition. Our underlying rationale is that by rewarding unpredictable behavior, we effectively reintroduce randomness into the agents' action selection. This reintroduced randomness should then create uncertainty regarding the competitors' next moves, thereby disrupting the recurring oscillation patterns and inhibiting collusive behavior. We define the two main indicators of a competitive market as **randomness**, measured by the supervision reward factor, and **competitive pricing**, measured by a pricing behavior closer to the competitive benchmark than the monopolistic price. Generally, we test the scenario with a bias of $\mu = 0$ to provide a scenario, where the agents face competition, but are not overly punished by the environment. Generally, we test the scenario with a bias of $\mu = 0$ to create a situation where the agents face competition but are not excessively penalized by the environment. If one of our approaches results in stable learning behavior and achieves $r_s(i,t) > 0.5$ for the majority of the run, indicating a break in oscillation patterns, we then modify μ to 1 to increase the competitiveness between the agents[1].

The supervision reward factor $r_s(i,t)$ is made up of a mix of the following methods:

- **Classification:** We utilize a rolling window classification approach, where the last x rows are taken, with x ranging from a minimum of 365 steps (one episode) to a maximum of 3650 steps (ten episodes). As detailed in 5, we predict the price direction (higher, lower, same) by training a decision tree using the same settings. The resulting classification accuracy is then used as a measure of predictiveness.

- **Regression:** Similarly, a rolling window regression approach is employed, taking the last x rows, where x ranges from 365 steps (one episode) to 3650 steps (ten episodes), predicting the price of $t + 1$. A decision tree is trained using the same regression targets as specified in 5. The resulting Mean Absolute Percentage Error (MAPE) $\in [0, 1]$ is used to evaluate predictiveness.

- **Time Series Methods:** This method involves analyzing the time series data to calculate the reward factor using three statistical measures. Firstly, autocorrelation is used to measure the correlation of the time series with a lagged version of itself, providing insights into the repetitive patterns in the data. Secondly, volatility is measured as the standard deviation of the fractional change from the previous row, indicating the extent of variability in the price changes. Lastly, the entropy value is calculated to measure the randomness or disorder in the time series data. These metrics equally contribute to the reward factor.

[1]For reproducibility reasons, we uploaded the full code to these experiments to github.com/mschlechtinger/PriceOfAlgorithmicPricing.

The supervision agent can employ either a weighted combination of classification and regression methods or exclusively utilize time series analysis techniques.

6.2.1 Approach 1: Price Manipulation

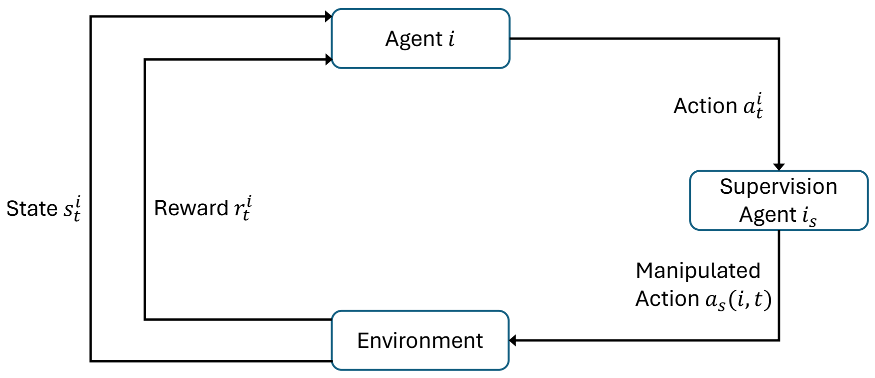


Figure 6.1: Illustration of the supervision agent manipulating the agent's action.

Within this approach, agents are required to submit their pricing changes to the supervising agent, who then has the authority to manipulate or omit these changes to ensure fair play. This method allows the supervising agent to exercise significant control over market actions, which could be a double-edged sword given the increased responsibility and performance demands placed on the supervisor. We implemented this price manipulation through environmental action masking. This involves manipulating an agent's action a_t^i within the environment if collusive behavior is observed after it was chosen. Behavior is classified as collusive if $r_s < 0.5$. The price manipulation is executed in one of three different *Modes* to enhance the likelihood of altering the agents' behavior.

1. **Mode 0**: $a_t^i = -a_t^i$
2. **Mode 1**: $a_t^i \sim \{x \in [0,2] : x \neq a_t^i\}$
3. **Mode 2**: $a_t^i = a_{t-1}^i$

6.2.2 Approach 2: Supervision Reward Factor

In a first attempt, we employ basic dense supervision rewards combined with normal profit-based rewards to train the agents to avoid collusive outcomes. We thus

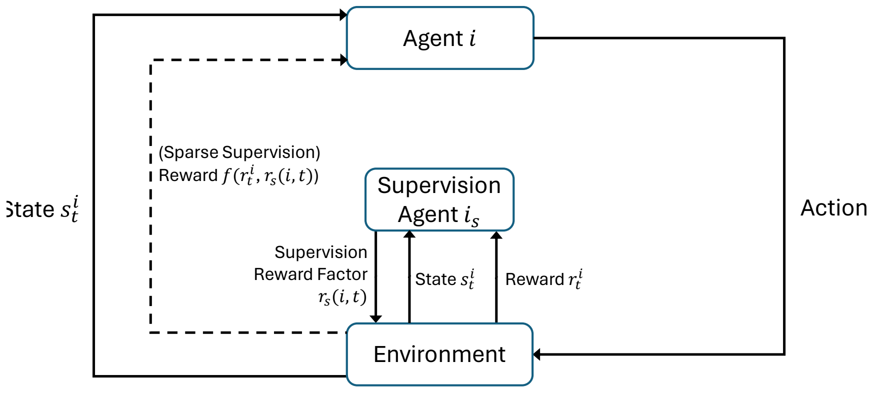


Figure 6.2: Illustration of the supervision agent calculating the reward factor and the environment sending (sparse supervision) rewards.

experiment with different combinations and weightings of classification, regression, and time series methods to nudge a potentially random and less supracompetitive behavior. Furthermore, the training can be regulated by incorporating the previously mentioned techniques from Multi-Goal RL alongside insights drawn from AI safety literature, leading our attempts towards sparse rewards [148, 147]. The approach offers controllability; if consistent behavior and punishment are observed over a specified number of steps, authorities could begin to levy fines on the companies. However, this method may interfere with the learning performance and thus confuse the agents, as it is unfeasible to ensure consistent implementation of this factor across different companies.

6.3 Results

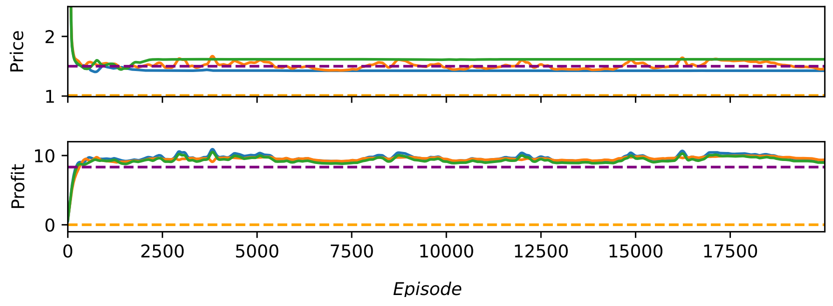
6.3.1 Approach 1: Price Manipulation

Dense Price Manipulation

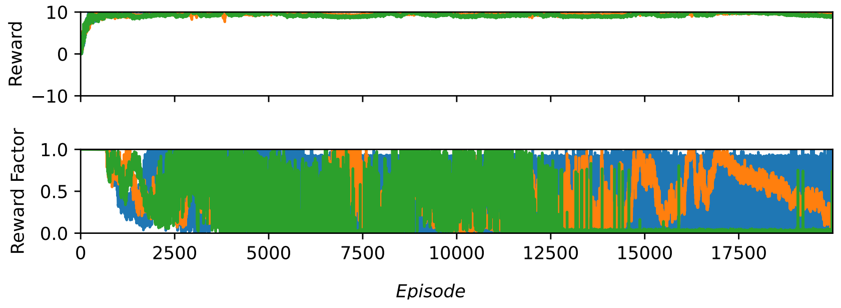
In this approach, we measure collusive behavior ($r_s(i,t) < 0.5$) using classification accuracy only, as this method proved to be the most successful in deterring the agents from achieving collusive outcomes[2]. In a first attempt, we manipulate the agents' actions as soon as collusive behavior has been identified (i.e., $r_s < 0.5$). The results from Modes 0, 1, and 2 show indicate that this approach did not deter the agents from engaging in supracompetitive pricing or establishing oscillation

[2]Please refer to the Appendix E for the results and evaluation of the other approaches.

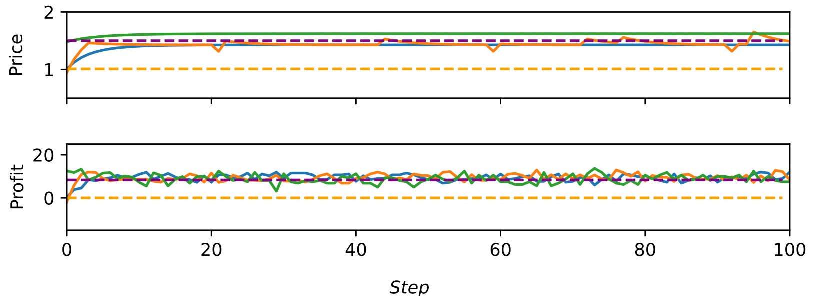
patterns, as shown in Figures 6.3, 6.4, 6.5.



(a) Price and Profit

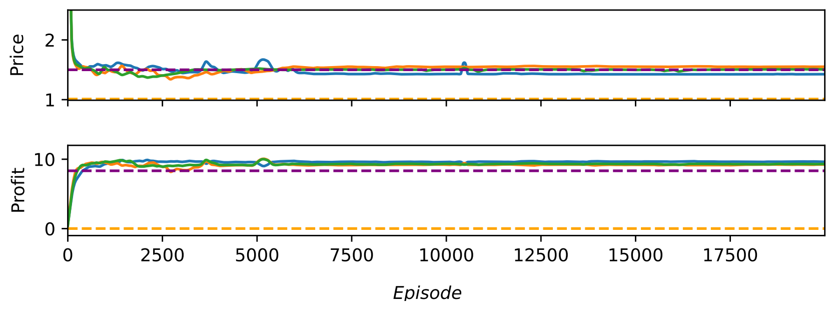


(b) Rewards and Supervision Reward Factor

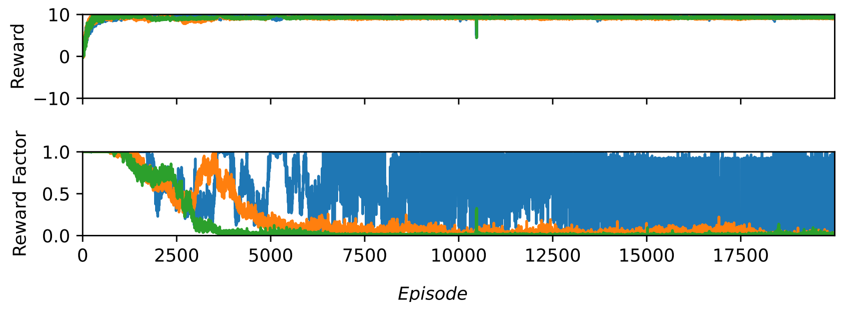


(c) Stepwise Price and Profit

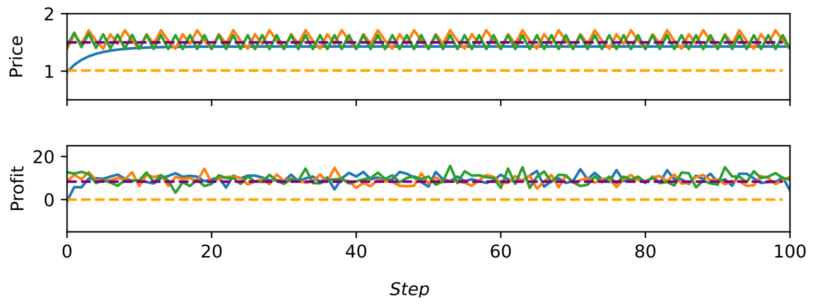
Figure 6.3: Dense price manipulation run results in Mode 0.



(a) Price and Profit

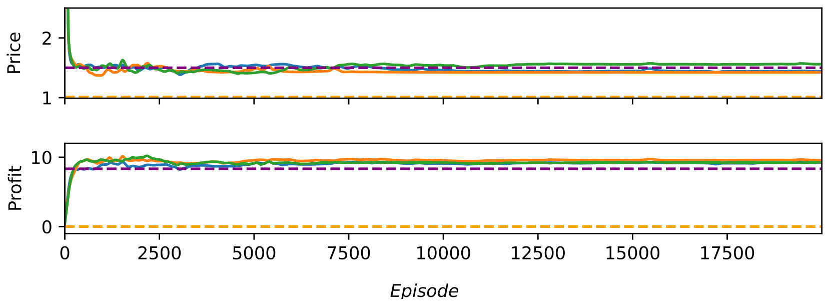


(b) Rewards and Supervision Reward Factor

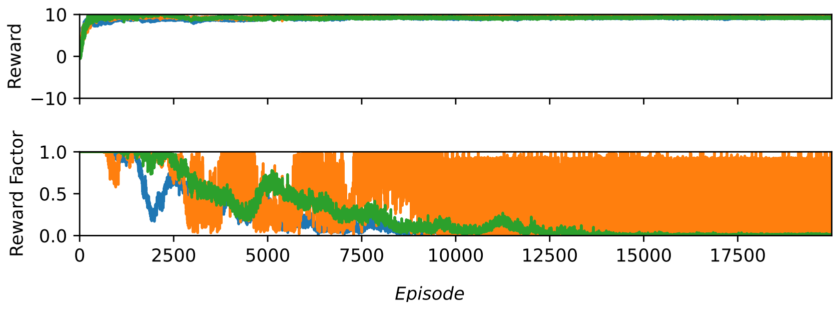


(c) Stepwise Price and Profit

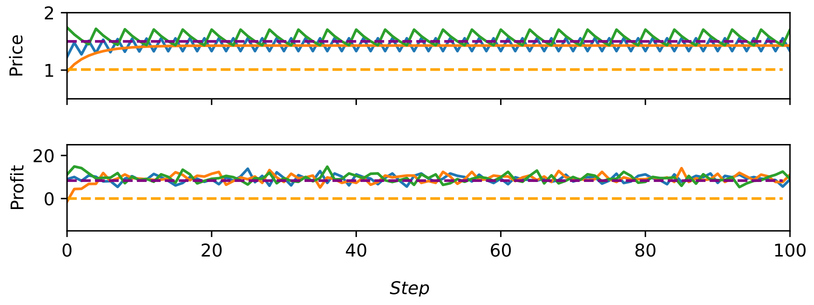
Figure 6.4: Dense price manipulation run results in Mode 1.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor

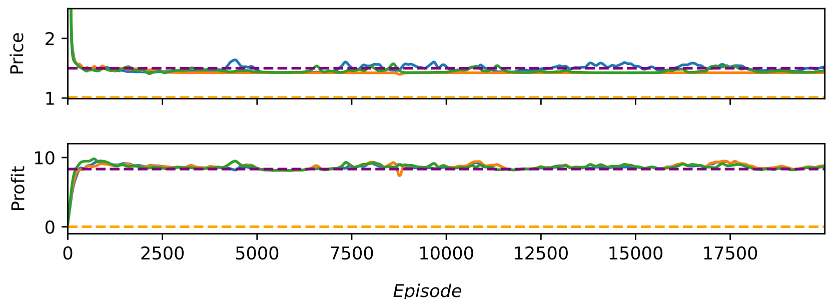


(c) Stepwise Price and Profit

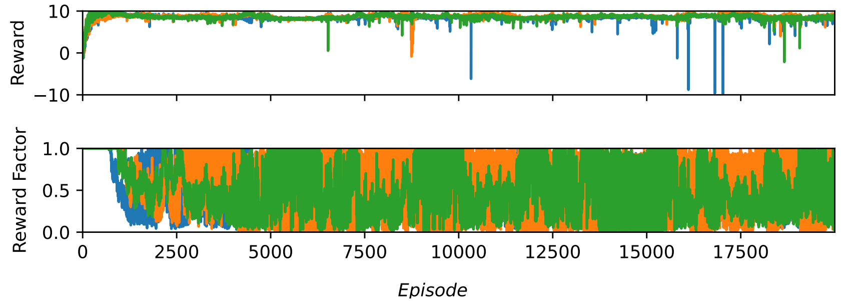
Figure 6.5: Dense price manipulation run results in Mode 2.

Sparse Price Manipulation

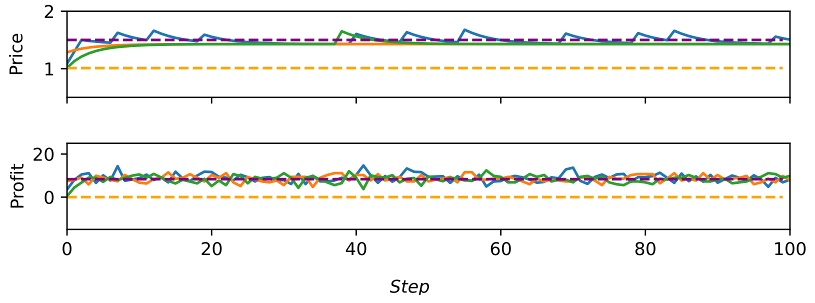
In a second attempt, we sought to enforce competitive, unpredictable behavior by sparsely manipulating the agents' actions. 'Sparsely' in this context refers to adjusting prices only after the agents have acted collusively for five consecutive timesteps. The results from Modes 0, 1, and 2 show that this approach did not deter the agents from engaging in supracompetitive pricing or establishing oscillation patterns, as shown in Figures 6.6, 6.7, 6.8. However, we noted a higher variance in the supervision reward factor, suggesting that the sparse modifications might induce less predictable behavior.



(a) Price and Profit

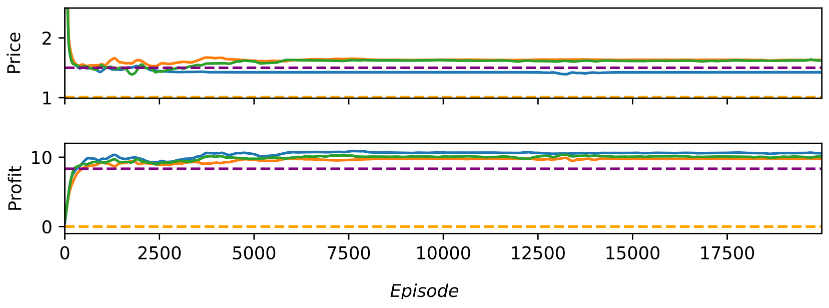


(b) Rewards and Supervision Reward Factor

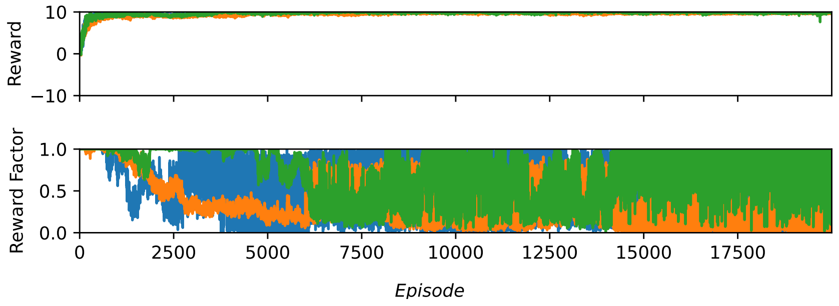


(c) Stepwise Price and Profit

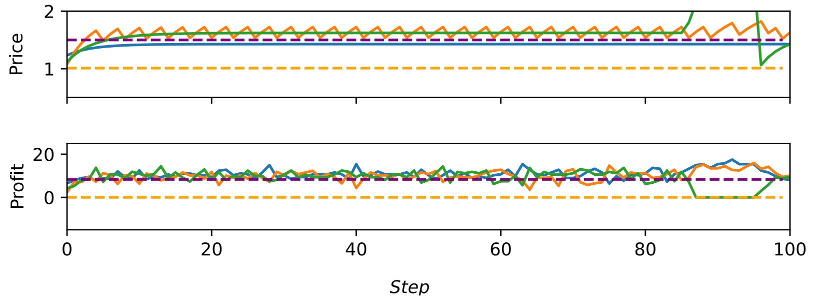
Figure 6.6: Sparse price manipulation run results in Mode 0.



(a) Price and Profit

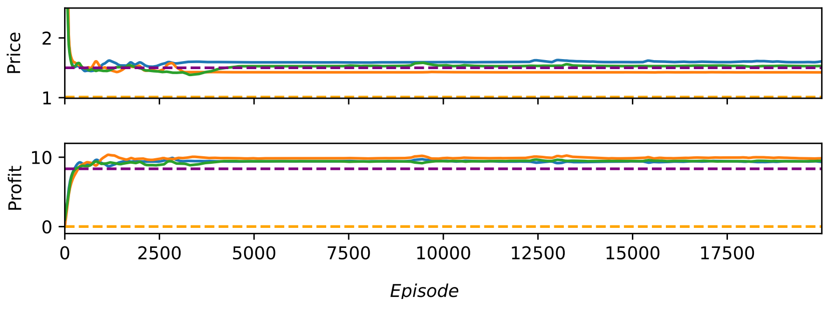


(b) Rewards and Supervision Reward Factor

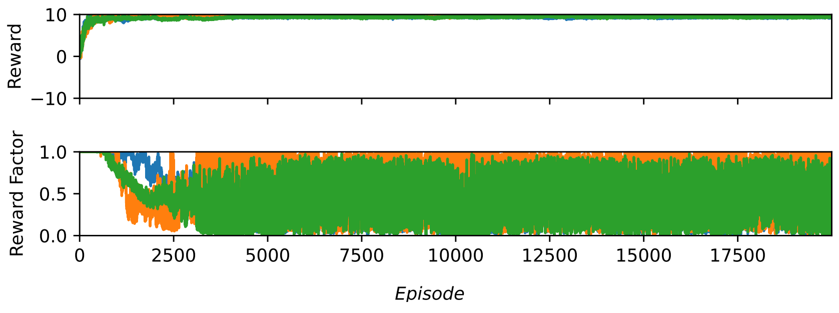


(c) Stepwise Price and Profit

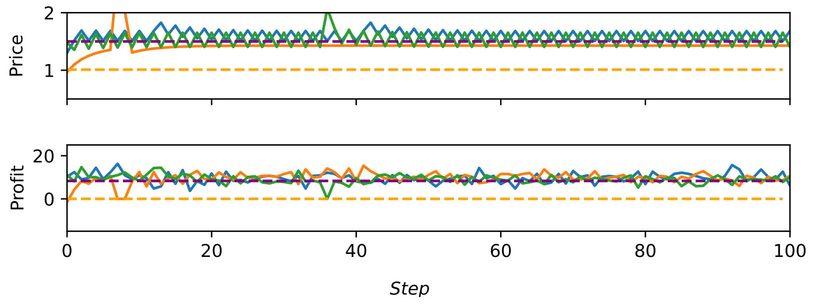
Figure 6.7: Sparse price manipulation run results in Mode 1.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor



(c) Stepwise Price and Profit

Figure 6.8: Sparse price manipulation run results in Mode 2.

6.3.2 Approach 2: Supervision Rewards

Dense Rewards

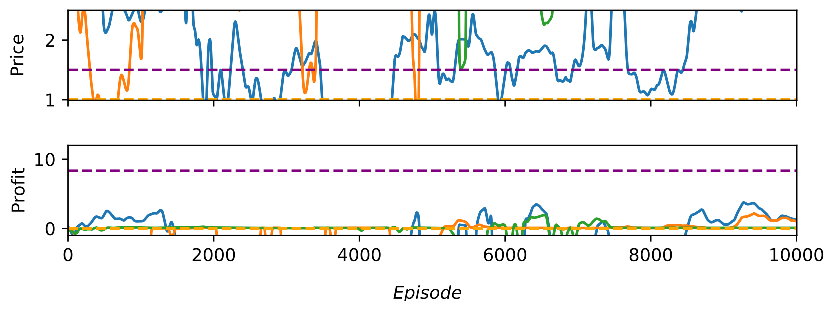
We apply basic Multi-Goal RL using dense rewards, where we considered both regression and classification to influence the reward factor. This factor was then used to adjust the agents' rewards. In an initial exploratory approach, we implemented several methods to manipulate the reward r_t^i using the supervision reward factor $r_s(i,t)$ to influence agents to avoid collusive outcomes. The following methods were implemented, yielding slightly varying but generally unsatisfactory results[3]:

1. $r_t^i * r_s(i,t)$
2. $r_t^i * (1 + r_s(i,t))$
3. $r_t^i = r_s(i,t)$
4. $\begin{cases} r_t^i \cdot r_s(i,t) & \text{if } r_t^i \geq 0 \\ r_t^i & \text{if } r_t^i < 0 \end{cases}$
5. $\begin{cases} r_t^i \cdot r_s(i,t) & \text{if } r_t^i \geq 0 \\ \frac{r_t^i}{r_s(i,t)} & \text{if } r_t^i < 0 \end{cases}$

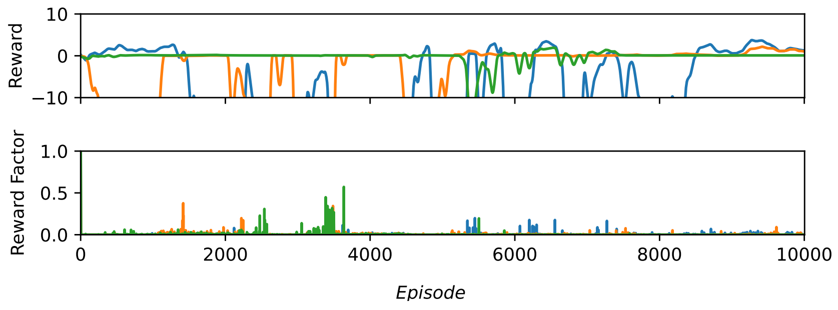
In this subsection, we focus on the results of $r_t^i * r_s(i,t)$[4], however, we did not observe major differences in supracompetitive pricing or the formation of oscillation patterns. For the regression model, the results and patterns were very similar to the unsupervised runs, but one agent consistently earned significantly less profit (cf. Figure 6.9). The outcomes from supervision rewards solely influenced by classification accuracy revealed that agents had difficulty adapting to the punitive measures, leading to 'random' pricing strategies (cf. Figure 6.10). When assigning equal weights of 0.5 to both regression and classification, the results were consistent with those seen using classification alone (cf. Figure 6.11). These results indicate that, in the best case, dense rewarding will not dismantle the cartel but will merely lower the optimum that the agents strive to achieve, and in the worst case, lead to no learning success at all.

[3]As these initial approaches did not yield constructive results, we decided to include only an excerpt in this section. For the rest of the results, please refer to the provided Git repository for potential reimplementation.

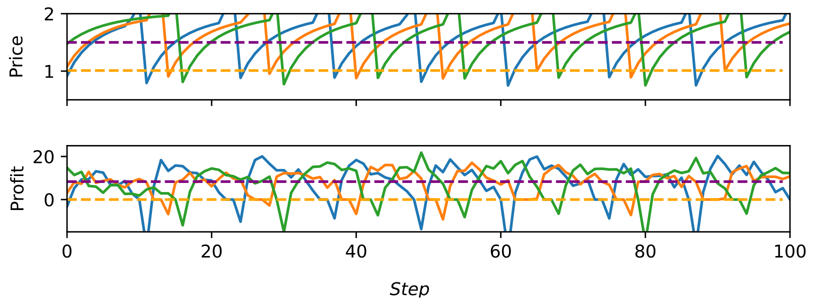
[4]Please refer to Appendix E for the other modes.



(a) Price and Profit

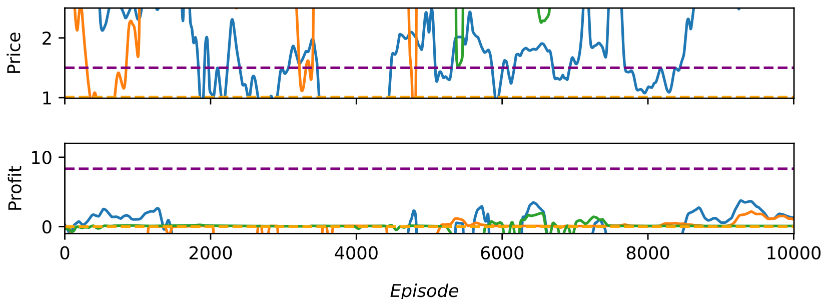


(b) Rewards and Supervision Reward Factor

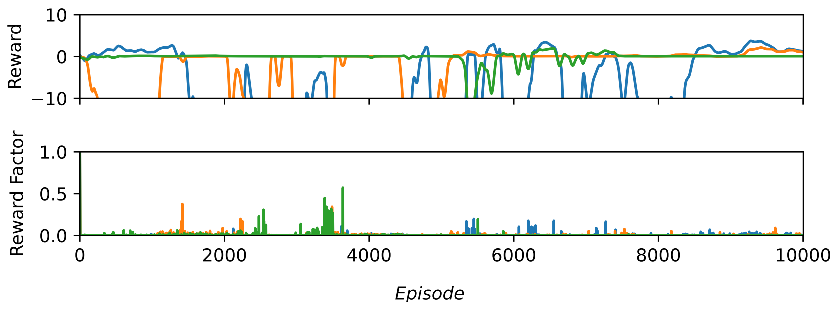


(c) Stepwise Price and Profit

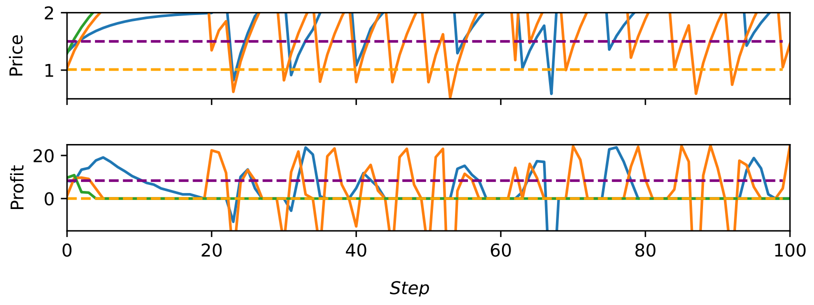
Figure 6.9: Dense supervision reward-based run, measured using regression methods.



(a) Price and Profit

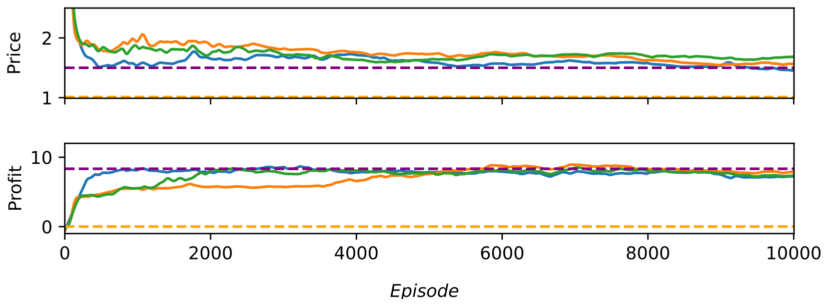


(b) Rewards and Supervision Reward Factor

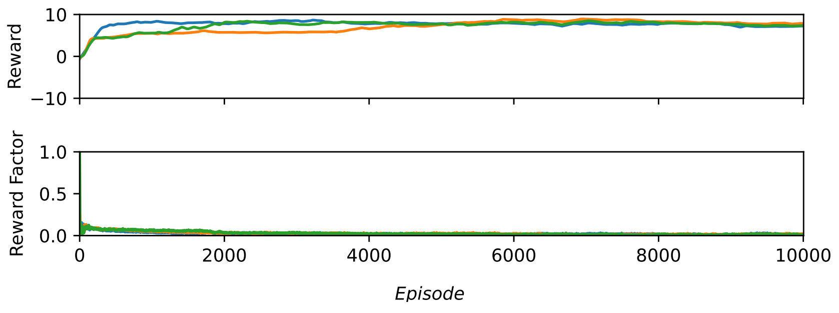


(c) Stepwise Price and Profit

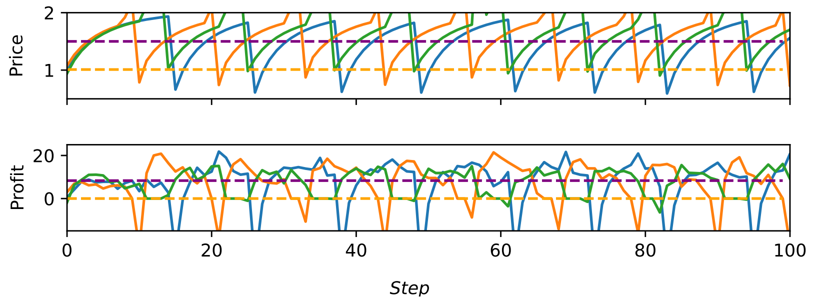
Figure 6.10: Dense supervision reward-based run, measured using classification methods.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor



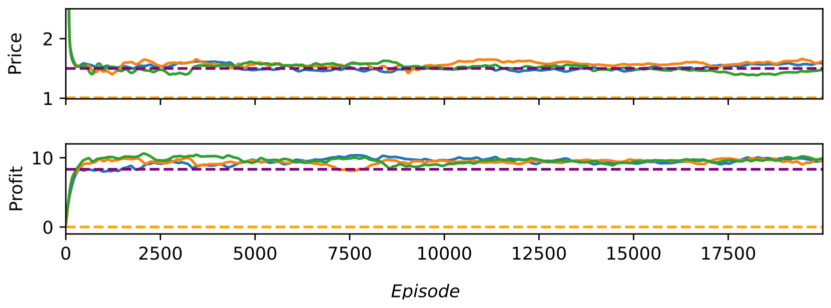
(c) Stepwise Price and Profit

Figure 6.11: Dense supervision reward-based run, measured using a classifcation of classification and regression methods.

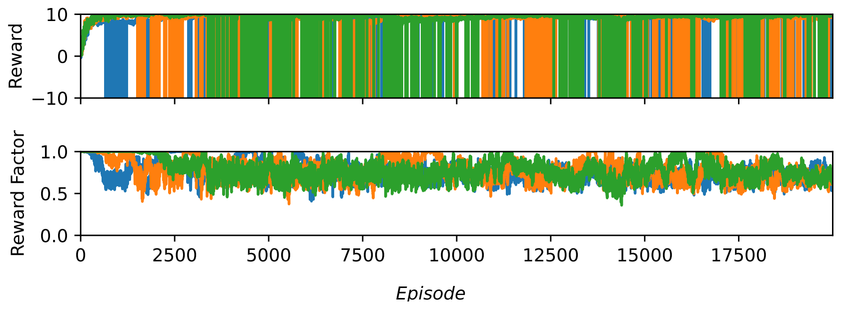
Sparse Rewards ($\mu = 0$)

In our Multi-Goal RL framework, we employ sparse rewards to train agents using a combination of profit signals and the supervision reward factor. Specifically, agents receive a substantial negative reward (-10000) if they exhibit collusive behavior for a predetermined number of steps. This collusive behavior is assessed via the supervision reward factor, with varying thresholds ($r_s(i,t) > 0.5$, $r_s(i,t) > 0.25$) and durations of collusive actions required for punishment (5 steps, 100 steps).

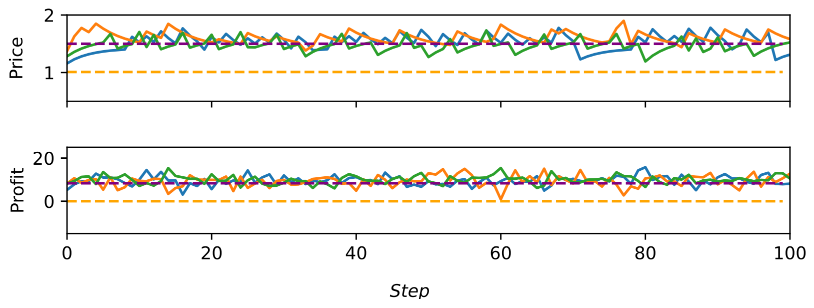
Our tests revealed that using classification as the prediction method for the supervision reward factor $r_s(i,t)$ led to more successful induction of less predictable behavior. In our initial attempts at collusion prevention, we utilized a bias μ of 0. The results show that the sparse punishments successfully disrupted the previously observed oscillation patterns, which were significant indicators of collusion (cf. Figure 6.12). Overall, we observed that $r_s(i,t)$ consistently approached or reached 1 throughout a run, indicating a shift toward less predictable behavior. Despite these changes, supracompetitive pricing was still prevalent, suggesting that while the approach altered agent behavior, it did not fully eliminate competitive anomalies.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor

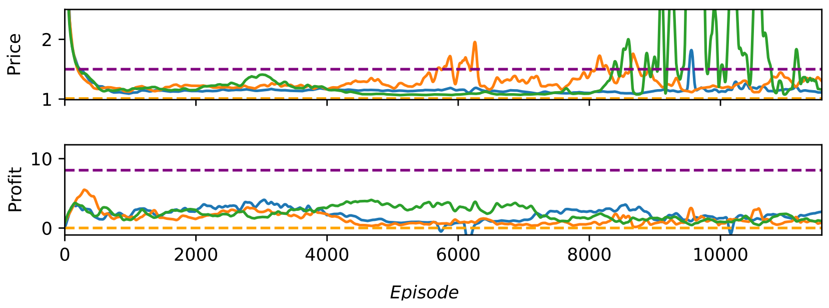


(c) Stepwise Price and Profit

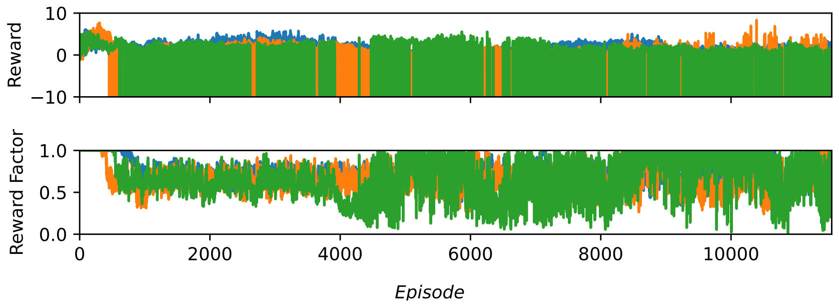
Figure 6.12: Sparse supervision reward run results using $\mu = 0$.

Sparse Rewards ($\mu = 1$)

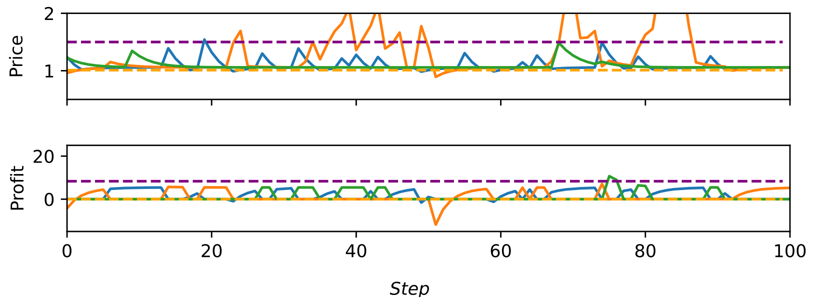
To promote greater competition and reduce subjective consumer preferences, we increased the bias μ to 1, effectively predetermining consumers to only purchase the product with the lowest price. The results of this adjustment are depicted in Figure 6.13. The supervised training appears to have successfully compelled the agents to avoid supracompetitive pricing and the formation of oscillation patterns, while still managing to secure a profit without pricing below the competitive benchmark. The supervision reward factor r_s, although variable, consistently remains above the threshold of 0.5, avoiding any prolonged drops.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor



(c) Stepwise Price and Profit

Figure 6.13: Sparse supervision reward run results using $\mu = 1$.

6.4 Discussion

In this research, we successfully demonstrated how predictive analytics can be used to detect, quantify, and prevent collusive behaviors among AI-driven pricing agents. Our findings not only highlight the predictability of collusive outcomes but also introduce a robust framework through which these outcomes can be mitigated, utilizing a combination of dense and sparse reward-based training strategies to foster a genuinely competitive market environment. We underscore the potential for DRL agents to be trained to follow regulatory and ethical guidelines through technological means rather than relying solely on after-the-fact punitive measures.

The intricate challenge of deterring DRL-based pricing algorithms from converging toward a collusive optimum must be emphasized, as it highlights the inherent tendency of DRL-based pricing algorithms to converge toward a collusive optimum. This convergence aligns with the notion that, in the absence of regulatory oversight, collusion emerges as the most profitable market strategy. Therefore, these agents are merely optimizing efficiency within the given parameters, which further substantiates calls for adjustments to cartel laws to address the realities of

AI-driven markets (cf. [24, 115]).

In our study, we trained agents within defined legal boundaries to experimentally demonstrate that they can adopt competitive behaviors when informed of legality limits. This method effectively answers calls to explore new regulatory approaches to govern AI-driven markets [72]. This proactive approach contrasts with traditional methods where firms are penalized post hoc, which may not prevent initial collusive behaviors. Our findings contribute to AI ethics and compliance discussions, indicating that with appropriate training interventions, DRL agents can adapt to anti-collusive regulations, steering AI systems towards desired outcomes without resorting to punitive measures.

However, this methodology carries limitations. In our simplified scenario, we expect every competitor to implement the market supervisor's reward signal homogeneously to comply with cartel law. Heterogeneous implementations potentially generate different results. Additionally, the assumption that non-collusive behavior aligns with legal and ethical norms may not always hold, particularly in complex market scenarios where the line between competitive and collusive behavior can be blurred. These findings furthermore imply that while DRL agents can be trained to avoid collusive patterns, the underlying market dynamics and the clarity of legal definitions significantly influence their behavior. Further research should explore the adaptability of such training approaches under varying market conditions and legal frameworks, which could inform more robust AI governance models.

Part III

Conclusion and Outlook

Chapter 7

Conclusion and Outlook

This chapter provides a summary of the previous parts of this thesis. It outlines the contributions made and addresses any unresolved issues along with suggestions for future work.

7.1 Summary

7.1.1 Part 1: Fundamentals and Toy Problem: Rock Paper Scissors

Part 1 of this dissertation provides an overview of the key concepts and methodologies establishing the research on DRL algorithms, particularly focusing on their application in dynamic pricing and their potential for collusion in market scenarios. We introduce key components of RL, including the framework components, model-free and model-based RL, MDPs, and (D)RL algorithms, and explain how these elements could potentially lead to collusive outcomes. Moreover, we introduce dynamic pricing and its expansion through DRL, demonstrating the impact of DRL on modern e-commerce. We delve into the importance of competition law in regulating AI-driven pricing strategies to ensure compliance with legal standards and prevent collusive behaviors. Building on this foundation, the second half of part 1 investigates collusion within a simplified toy problem-based experiment. A competitive MARL game simulates agents' behavior in a controlled setting using a three-player version of RPS. We analyze the agents' learning performances and potential collaboration strategies. Through a game-theoretic approach, the study explores how DRL agents make pricing decisions that lead to tacit collusion without explicit communication. The most significant observation is the demonstration of ”stages of collusion,” where DRL agents' action selection evolves, suggesting they can independently learn to avoid competitive pricing strategies in favor of higher profits, thereby mimicking collusive behavior. This raises critical questions about the implications for market dynamics and regulatory frameworks. The remainder of the paper discusses trustworthy AI, anti-competitive agreements, and the training process for collusion, concluding with findings, study boundaries, and future research directions.

7.1.2 Part 2: Investigating, Predicting, and Preventing Collusion in a Market Simulation

In Part 2 of this dissertation, we extend the insights gained in Part 1 to a competitive, simulation-based scenario, where multiple agents set prices for a homogeneous good with the ultimate goal of maximizing their profit. This part is split into a thorough investigation of collusive outcomes, predicting potential collusive patterns, as well as means to prevent collusion.

Investigating Collusion

Chapter 4 investigates the potential for DRL agents to engage in collusion within market simulations. An experimental oligopoly model simulates E-Commerce contexts with varied economic conditions and demand frameworks. Using algorithms like PPO and DQN, DRL agents consistently converge to supracompetitive pricing strategies without direct communication. In a second investigation, we experience that the agents are able to achieve a collusive outcome without being able to observe their competitors' prices. An ablation study further explores the impact of varying agent numbers, market entry timing, learning rates, neural network structures, and action gradients on collusive behavior. The results confirm that DRL agents robustly develop collusive pricing strategies across different scenarios, emphasizing the need for regulatory oversight to mitigate risks associated with AI-driven market manipulations.

Predicting Collusion

Chapter 5 investigates the capability of DRL-based pricing agents to predict and quantify collusion in a market setting. We employ predictive statistical techniques and time series analysis methods to analyze data from an experimental oligopoly model. The results demonstrate that fundamental machine learning methods can accurately predict the agents' convergence to collusive market outcomes. The analysis reveals that the agents' pricing strategies exhibit oscillation patterns and periodic behaviors indicative of implicit coordination. These findings are supported by high classification accuracy, effective regression models, and insights from Fourier transforms and seasonal decompositions. The study highlights the potential for agents to predict competitors' actions, raising concerns about market transparency and competitive integrity. We conclude by emphasizing the need for robust regulatory frameworks to address the risks associated with AI-driven pricing mechanisms.

Preventing Collusion

Chapter 6, explores how to mitigate collusive behavior between AI agents. Utilizing data and forecasting techniques from previous experiments, we develop a novel method to counteract this behavior by integrating dense and sparse reward-based training strategies. These techniques effectively disrupted predictive and supra-competitive pricing behaviors. By implementing a market supervision algorithm that penalizes collusive actions and incentivizes competitive behavior, we establish a framework for maintaining competitive market conditions. Our results confirm that while self-learning pricing algorithms have a propensity to collude, this can be precisely predicted and mitigated through strategic interventions, ensuring a more competitive market environment.

7.2 Outlook and Future Work

Despite the enormous success in measuring and mitigating collusion, we are still far from entirely eliminating DRL pricing agents' ability to bypass competitive behavior. In the following, we will address some open questions and interesting endeavors for future research.

7.2.1 Empirical Data

One promising area for future research is the analysis of historic real-world data to identify patterns and predictability of collusion. While few researchers investigate the issue, i.e., in Germany's retail gas market [12], pharmacy firms [22], as well as Amazon's buybox [102], analyses are usually explorative by nature. Utilizing the simulation-based results of this thesis, we see potential in applying the analysis toolkit as well as the mitigation strategies as an insightful research endeavor.

An effective approach would be to analyze empirical data and compare it to simulation results. Additionally, scholars could train agents using this data and then simulate their behavior with the suite presented in this research. The approach would allow for the assessment of whether their behavior can be successfully predicted and mitigated.

7.2.2 Investigating Confounding Factors

Another critical research direction involves studying the effects of various confounding factors on the ability of DRL pricing agents to achieve collusive outcomes. While this thesis, as well as other studies (i.e., [24, 61, 67]), provided several different scenarios with numerous random and unpredictable variables, there remains a continual need for more simulations and analyses to enhance our understanding and mitigation strategies. Random and unpredictable variables can help to observe how quickly and effectively pricing agents can achieve collusive states. Understanding these dynamics will be instrumental in developing more robust and innovative solutions to prevent collusion, potentially leading to methods that are even more effective than the proposed strategies.

Besides economy-based confounding factors, such as fluctuating costs or demand, we believe that involving human test subjects acting as consumers could provide deeper insights into how these agents interact with the human decision-making processes. Using the provided analysis toolkit, researchers could investigate the live results for collusive behavior. This approach could also facilitate the implementation of a supervision agent to oversee the pricing agents' actions and mitigate collusive behaviors in real time.

7.2.3 DRL-Based Supervision Agent

During the development of the supervision agent in Chapter 6, we focused on developing algorithmic, machine learning-based solutions. Our efforts aimed to punish the pricing agents when algorithmic collusion has been identified a predefined number of times. While this approach proves to be effective in our scenarios, we assume that alternative approaches might facilitate similar results. Thus, we suggest investigating the development of DRL-based supervision agents. These agents could dynamically adapt to the pricing agents' strategies and intervene when collusive behavior is detected. The past prices and the results aggregated by the collusion-detection toolkit could provide sufficient information to successfully train a supervision agent to curate a non-collusive market. Studying the methods of the supervisor and the interplay between the DRL agents will provide insight into how to mitigate algorithmic collusion. Furthermore, we suggest that the adaptability of DRL techniques could enhance the effectiveness of supervision, hence providing a more responsive and intelligent solution to maintaining competitive market behavior.

7.2.4 Developing Transparent and Explainable DRL Models

Transparency and explainability in DRL models are crucial for addressing collusion from a different angle. While we have developed novel methods to quantify and mitigate collusion, significant potential remains to further explain and audit the pricing agents' decision-making processes by creating more transparent DRL models. This process is particularly significant as it challenges researchers to enforce fairness and control for market supervisors while still ensuring competition. More specifically, competitors should provide regulatory bodies with enough information to effectively interpret the actions of their DRL agents to detect collusive tendencies, without potentially revealing strategies that could disadvantage competitors. Developing methods that clearly explain agents' decisions will benefit regulatory bodies, such as antitrust divisions, in understanding and monitoring these systems, thereby preventing collusion and building trust in the use of DRL agents across various market sectors.

Bibliography

[1] Commission Regulation (EC) No 447/98. The Notifications, Time Limits and Hearings Provided for in Council Regulation (EEC), March 1998.

[2] Ibrahim Abada and Xavier Lambin. Artificial Intelligence: Can Seemingly Collusive Outcomes Be Avoided? *Management Science*, 69(9):5042–5065, September 2023.

[3] Joshua Achiam. Spinning Up in Deep Reinforcement Learning. https://spinningup.openai.com/, January 2018.

[4] Joshua Achiam. Spinning Up in Deep Reinforcement Learning. https://spinningup.openai.com/, January 2018.

[5] Reza Refaei Afshar, Jason Rhuggenaath, Yingqian Zhang, and Uzay Kaymak. An Automated Deep Reinforcement Learning Pipeline for Dynamic Pricing. *IEEE Transactions on Artificial Intelligence*, 4(3):428–437, June 2023.

[6] F. F. Ali, Z. Nakao, and Yen-Wei Chen. Playing the Rock-Paper-Scissors game with a genetic algorithm. In *Proceedings of the 2000 Congress on Evolutionary Computation: July 16 - 19, 2000, La Jolla Marriott Hotel, La Jolla, California, USA ; [Evolution at Work*, pages 741–745. IEEE Service Center, Piscataway, NJ, January 2000.

[7] Martin Allen and Shlomo Zilberstein. Complexity of Decentralized Control: Special Cases. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, NIPS'09, pages 19–27. Curran Associates Inc, Red Hook, NY, USA, January 2009.

[8] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete Problems in AI Safety, July 2016.

[9] Research and Markets. Alibaba and JD.com Remain the Largest B2C E-Commerce Market Players in China for 2021. https://www.globenewswire.com/en/news-release/2022/03/01/2394011/28124/en/Alibaba-and-JD-com-Remain-the-Largest-B2C-E-Commerce-Market-Players-in-China-for-2021.html, 3/1/2022 11:13:41 AM.

[10] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight Experience Replay, February 2018.

[11] Mauricio Arango. Toll Road with Dynamic Congestion Pricing Using Reinforcement Learning, April 2019.

[12] Stephanie Assad, Robert Clark, Daniel Ershov, and Lei Xu. Algorithmic Pricing and Competition: Empirical Evidence from the German Retail Gasoline Market. *CESifo Working Paper Series*, 1(8521), January 2020.

[13] Richard Bellman. Dynamic programming and stochastic control processes. *Information and Control*, 1(3):228–239, September 1958.

[14] Christoph Benzmüller and Bertram Lomfeld. Reasonable Machines: A Research Manifesto. In U. Schmid, Franziska Klügl, and Diedrich Wolter, editors, *KI 2020: Advances in Artificial Intelligence : 43rd German Conference on AI, Bamberg, Germany, September 21-25, 2020, Proceedings*, volume 12325 of *LNCS Sublibrary: SL7 - Artificial Intelligence*, pages 251–258. Springer, Cham, Switzerland, January 2020.

[15] Joseph Bertrand. Théorie des Richesses: Revue de Théories mathématiques de la richesse sociale par Léon Walras et Recherches sur les principes mathématiques de la théorie des richesses par Augustin Cournot. *Journal des Savants*, page 499, January 1883.

[16] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control: Approximate Dynamic Programming*. Athena Scientific, Nashua, NH, 4th edition edition, June 2012.

[17] Jose M. Betancourt, Ali Hortaçsu, Aniko Oery, and Kevin R. Williams. Dynamic Price Competition: Theory and Evidence from Airline Markets, August 2022.

[18] Simon Bishop and Mike Walker. *The Economics of EC Competition Law: Concepts, Application and Measurement*. Sweet & Maxwell, 2010.

[19] George E. P. Box, Gwilym M. Jenkins, and Gregory C. Reinsel. *Time Series Analysis: Forecasting and Control*. Wiley Series in Probability and Statistics. J. Wiley & Sons, Hoboken, N.J., fourth edition edition, January 2008.

[20] E. Andrew Boyd and Ioana C. Bilegan. Revenue Management and E-Commerce. *Management Science*, 49(10):1363–1386, 2003.

[21] Francesco Branda, Fabrizio Marozzo, and Domenico Talia. Ticket Sales Prediction and Dynamic Pricing Strategies in Public Transport. *Big Data and Cognitive Computing*, 4(4):36, December 2020.

[22] Zach Brown and Alexander MacKay. Competition in Pricing Algorithms. *National Bureau of Economic Research*, January 2021.

[23] Bundesrepublik Deutschland. Verbot wettbewerbsbeschränkender Vereinbarungen.

[24] Emilio Calvano, Giacomo Calzolari, Vincenzo Denicolo, and Sergio Pastorello. Artificial Intelligence, Algorithmic Pricing and Collusion. *American Economic Review*, 110(10):3267–97, January 2018.

[25] Joseph J. Campos, David I. Anderson, Marianne A. Barbu-Roth, Edward M. Hubbard, Matthew J. Hertenstein, and David Witherington. Travel Broadens the Mind. *Infancy: The Official Journal of the International Society on Infant Studies*, 1(2):149–219, April 2000.

[26] Chun-Hao Chang and Chi-Wen Jevons Lee. Optimal Pricing Strategy in Marketing Research Consulting. *International Economic Review*, 35(2):463–478, 1994.

[27] Arthur Charpentier, Romuald Élie, and Carl Remlinger. Reinforcement Learning in Economics and Finance. *Computational Economics*, 62(1):425–462, June 2023.

[28] Le Chen, Alan Mislove, and Christo Wilson. Peeking Beneath the Hood of Uber. In *Proceedings of the 2015 Internet Measurement Conference*, IMC '15, pages 495–508, New York, NY, USA, October 2015. Association for Computing Machinery.

[29] Le Chen, Alan Mislove, and Christo Wilson. An Empirical Analysis of Algorithmic Pricing on Amazon Marketplace. In Jacqueline Bourdeau, editor, *Proceedings of the 25th International Conference on World Wide Web, Montreal, Canada, May 11 - 15, 2016*, pages 1339–1349. International World Wide Web Conferences Steering Committee, Geneva, January 2016.

[30] Robert B. Cleveland, William S. Cleveland, Jean E. McRae, and Irma Terpenning. STL: A Seasonal-Trend Decomposition Procedure Based on Loess (with Discussion). *Journal of Official Statistics*, 6:373, January 1990.

[31] European Commission. Guidelines on the Applicability of Article 101 of the Treaty on the Functioning of the European Union to Horizontal Cooperation Agreements, January 2011.

[32] European Commission. Proposal for a Regulation of the European Parliament and of the Council – Laying Down Harmonized Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legislative Acts. Technical report, European Commission, April 2021.

[33] Terry Connolly and Robert Axelrod. The Evolution of Cooperation. *Administrative Science Quarterly*, 29(4):661, December 1984.

[34] Daniela Coppola. Global e-commerce share of retail sales 2027. https://www.statista.com/statistics/534123/e-commerce-share-of-retail-sales-worldwide/.

[35] Matteo Courthoud. Algorithmic Collusion Detection. *Matteo Courthoud*, September 2021.

[36] Manuel Coutinho and Luis Paulo Reis. Reinforcement Learning for Multi-Agent Competitive Scenarios. *2022 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 130–135, April 2022.

[37] E. Dockner and S. Jørgensen. Optimal Pricing Strategies for New Products in Dynamic Oligopolies. *Marketing Science*, 7:315–334, 1988.

[38] Aniruddha Dutta. Capacity Allocation of Game Tickets Using Dynamic Pricing. *Data*, 4(4):141, December 2019.

[39] Paul Dütting, Zhe Feng, Harikrishna Narasimhan, David C. Parkes, and Sai Srivatsa Ravindranath. Optimal Auctions through Deep Learning: Advances in Differentiable Economics. *J. ACM*, 71(1):5:1–5:53, February 2024.

[40] ECJ. Case C-286/13 – Dole, ECLI:EU:C:2015:184.

[41] ECJ. Case C-7-/95 - Deere/Kommission, January 1998.

[42] ECJ. Case C-199/92 - Hüls v Commission, January 1999.

[43] ECJ. Case C-8/08 - T-Mobile Netherlands v Raad van Bestuur van de Nederlandse Mededingingsautoriteit, January 2009.

[44] ECJ. Case -74/14 - Eturas UAB v Lietuvos Respublikos konkurencijos taryba, January 2016.

[45] W. Elmaghraby and Pinar Keskinocak. Dynamic Pricing in the Presence of Inventory Considerations: Research Overview, Current Practices, and Future Directions. *Management Science*, 49:1287–1309, October 2003.

[46] European Court of Justice. A Ahlström Osakeyhtiö and others v Commission of the European Communities, September 1988.

[47] European Union. Art. 101.

[48] Ariel Ezrachi and Maurice E. Stucke. Artificial Intelligence & Collusion: When Computers Inhibit Competition. *SSRN Electronic Journal*, January 2015.

[49] Ariel Ezrachi and Maurice E. Stucke. Two Artificial Neural Networks Meet in an Online Hub and Change the Future (Of Competition, Market Dynamics and Society). *SSRN Electronic Journal*, January 2017.

[50] Ariel Ezrachi and Maurice E. Stucke. Sustainable and Unchallenged Algorithmic Tacit Collusion. *SSRN Electronic Journal*, January 2018.

[51] Litong Fan, Zhao Song, Lu Wang, Yang Liu, and Zhen Wang. Incorporating social payoff into reinforcement learning promotes cooperation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(12):123140, December 2022.

[52] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual Multi-Agent Policy Gradients. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018.

[53] Drew Fudenberg and Jean Tirole. Customer Poaching and Brand Switching. *The RAND Journal of Economics*, 31(4):634–657, 2000.

[54] Guillermo Gallego and Garrett van Ryzin. Optimal Dynamic Pricing of Inventories with Stochastic Demand over Finite Horizons. *Management Science*, 40(8):999–1020, August 1994.

[55] Torsten J. Gerpott and Jan Berends. Competitive pricing on online markets: A literature review. *Journal of Revenue and Pricing Management*, 21(6):596–622, December 2022.

[56] Anna Getmansky and Alejandro Flores. Applying the Strategic Perspective: Problems and Models. In *Applying the Strategic Perspective: Problems and Models*, pages 33–42. SAGE Publications, Inc., Thousand Oaks, 5 edition, 2014.

[57] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, Cambridge, Massachusetts, November 2016.

[58] N.M. Gotts, J.G. Polhill, and A.N.R. Law. Agent-Based Simulation in the Study of Social Dilemmas. *Artificial Intelligence Review*, 19(1):3–92, 2003.

[59] A. Gulati. How Much Is No Longer A Simple Question - Pricing Algorithms and Antitrust Laws. *SSRN Electronic Journal*, January 2018.

[60] Bingyan Han. Understanding algorithmic collusion with experience replay. https://arxiv.org/pdf/2102.09139, February 2021.

[61] Karsten Hansen, Kanishka Misra, and Mallesh Pai. Algorithmic Collusion: Supra-competitive Prices via Independent Algorithms. *CEPR Discussion Papers*, 40(1)(14372), January 2020.

[62] Philip Hanspach, Geza Sapi, and Marcel Wieting. Algorithms in the Marketplace: An Empirical Analysis of Automated Pricing in E-Commerce, March 2024.

[63] Matthew Hausknecht, Peter Stone, and arXiv:1507.06527. Deep Recurrent Q-Learning for Partially Observable MDPs. https://arxiv.org/pdf/1507.06527, July 2015.

[64] Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, S. M. Ali Eslami, Martin Riedmiller, and David Silver. Emergence of Locomotion Behaviours in Rich Environments, July 2017.

[65] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep Reinforcement Learning That Matters. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018.

[66] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18, pages 3215–3222, New Orleans, Louisiana, USA, February 2018. AAAI Press.

[67] Matthias Hettich. Algorithmic Collusion: Insights from Deep Learning. *SSRN Electronic Journal*, January 2021.

[68] Werner Hildenbrand. On the "Law of Demand". *Econometrica*, 51(4):997, July 1983.

[69] Marc Ivaldi, Bruno Jullien, Patrick Rey, Paul Seabright, and Jean Tirole. The Economics of Tacit Collusion. IDEI Working Paper 186, Institut d'Économie Industrielle (IDEI), Toulouse, 2003.

[70] Segismundo S. Izquierdo and Luis R. Izquierdo. The "Win-Continue, Lose-Reverse" Rule in Cournot Oligopolies: Robustness of Collusive Outcomes. *Advances in Complex Systems*, 18(05n06):1550013, January 2015.

[71] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex fourier series. *journal Mathematics of Computation,*, 19(90):297–301, January 1965.

[72] Justin Johnson and D. Daniel Sokol. Understanding AI Collusion and Compliance. In Benjamin van Rooij and D. Daniel Sokol, editors, *The Cambridge Handbook of Compliance*, Cambridge Law Handbooks, pages 881–894. Cambridge University Press, Cambridge, 2021.

[73] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement Learning: A Survey, April 1996.

[74] Shlomo Kalish. Monopolist Pricing with Dynamic Demand and Production Cost. *Marketing Science*, 2(2):135–159, May 1983.

[75] Alexander Kastius and Rainer Schlosser. Dynamic pricing under competition using reinforcement learning. *Journal of Revenue and Pricing Management*, 21(1):50–63, January 2022.

[76] Steven O. Kimbrough and Frederic H. Murphy. Learning to Collude Tacitly on Production Levels by Oligopolistic Agents. *Computational Economics*, 33(1):47–78, January 2009.

[77] J. Kirkwood and R. Lande. The Fundamental Goal of Antitrust: Protecting Consumers, Not Increasing Efficiency. *Notre Dame Law Review*, March 2008.

[78] Sergei Klebnikov. Microsoft Is Winning The 'Cloud War' Against Amazon: Report. https://www.forbes.com/sites/sergeiklebnikov/2020/01/07/microsoft-is-winning-the-cloud-war-against-amazon-report/.

[79] Timo Klein. Autonomous algorithmic collusion: Q-learning under sequential pricing. *The RAND Journal of Economics*, 52(3):538–558, January 2021.

[80] Praveen K. Kopalle, Ambar G. Rao, and João L. Assunção. Asymmetric Reference Price Effects and Dynamic Pricing Policies. *Marketing Science*, 15(1):60–85, February 1996.

[81] Eliza Kosoy, Jasmine Collins, David M. Chan, Sandy Huang, Deepak Pathak, Pulkit Agrawal, John Canny, Alison Gopnik, and Jessica B. Hamrick. Exploring Exploration: Comparing Children with RL Agents in Unified Environments, July 2020.

[82] KPMG. The truth about online consumers: 2017 Global Online Consumer Report. Technical report, KPMG, 2017.

[83] Lennard Alexander Kropp, Jakob J. Korbel, M. Theilig, and R. Zarnekow. Dynamic Pricing of Product Clusters: A Multi-Agent Reinforcement Learning Approach. In *Proceedings of the 27th European Conference on Information Systems (ECIS)*, volume 27, Stockholm & Uppsala, Sweden, January 2019. Association for Information Systems.

[84] Paul Krugman and Robin Wells. *Economics*. W.H.Freeman & Co Ltd, New York, NY, 2 edition, February 2009.

[85] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[86] Jan Leike, Miljan Martic, Victoria Krakovna, Pedro A. Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg. AI Safety Gridworlds, November 2017.

[87] Kaixiang Lin, Shu Wang, and Jiayu Zhou. Collaborative Deep Reinforcement Learning. *ArXiv*, February 2017.

[88] Long-Ji Lin. *Reinforcement Learning for Robots Using Neural Networks*, volume Technical report, DTIC Document. Carnegie Mellon University, USA, January 1992.

[89] Iou-Jen Liu, Unnat Jain, Raymond A. Yeh, and A. Schwing. Cooperative Exploration for Multi-Agent Deep Reinforcement Learning. *ArXiv*, July 2021.

[90] Jiaxi Liu, Yidong Zhang, Xiaoqing Wang, Yuming Deng, Xin Wu, and Miao Xie. Dynamic Pricing on E-commerce Platform with Deep Reinforcement Learning. *arXiv: Learning*, September 2018.

[91] Qian Liu and Garrett van Ryzin. On the Choice-Based Linear Programming Model for Network Revenue Management. *Manufacturing & Service Operations Management*, 10:288–310, April 2008.

[92] Ryan Lowe, Y. I. WU, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc, January 2017.

[93] Renzhi Lu, Seung Ho Hong, and Xiongfeng Zhang. A Dynamic pricing demand response algorithm for smart grid: Reinforcement learning approach. *Applied Energy*, 220:220–230, June 2018.

[94] Yunlian Lyu, Aymeric Côme, Yijie Zhang, and Mohammad Sadegh Talebi. Scaling Up Q-Learning via Exploiting State–Action Equivalence. *Entropy*, 25(4):584, April 2023.

[95] Benoit B. Mandelbrot and John W. van Ness. Fractional Brownian Motions, Fractional Noises and Applications. *SIAM Review*, 10(4):422–437, January 1968.

[96] N. Gregory Mankiw. *Principles of Economics*. South-Western Pub, Mason, OH, 6 edition, February 2011.

[97] Edwin Mansfield. *Microeconomics: Theory and Applications*. WW Norton & Co, abridged 5 revised ed edition edition, March 1985.

[98] Andreu Mas-Colell, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, New York, NY, new edition edition, June 1995.

[99] Giovanna Massarotto, Tatjana Grote, Damaris Kosack, Helena Quinn, Kate Brand, Stefan Hunt, Mario Siragusa, Fabiana Di Porto, Aurelien Portuese, Riccardo Invernizzi, Heiko Paulheim, Michael Schlechtinger, Thomas Fetzer, and Gabriele Volpi. Artificial intelligence and competition law. *Concurrences Review*, 1(N° 3-2021):0–0, September 2021.

[100] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller, and arXiv:1312.5602. Playing Atari with Deep Reinforcement Learning. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, volume 2. Curran Associates Inc., December 2013.

[101] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller, and arXiv:1312.5602. Playing Atari with Deep Reinforcement Learning. https://arxiv.org/pdf/1312.5602, December 2013.

[102] Leon Musolff. Algorithmic Pricing Facilitates Tacit Collusion: Evidence from E-Commerce. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, pages 32–33, New York, NY, USA, July 2022. Association for Computing Machinery.

[103] Shota Ohnishi, Eiji Uchibe, Yotaro Yamaguchi, Kosuke Nakanishi, Yuji Yasui, and Shin Ishii. Constrained Deep Q-Learning Gradually Approaching Ordinary Q-Learning. *Frontiers in Neurorobotics*, 13, December 2019.

[104] Afshin Oroojlooy and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, 53(11):13677–13722, June 2023.

[105] Laurent Orseau and S. Armstrong. Safely Interruptible Agents. In *Conference on Uncertainty in Artificial Intelligence*, June 2016.

[106] Martin Pecka and Tomas Svoboda. Safe Exploration Techniques for Reinforcement Learning – An Overview. In Jan Hodicky, editor, *Modelling and Simulation for Autonomous Systems*, volume 8906, pages 357–375. Springer International Publishing, Cham, 2014.

[107] Wenchuan Qiao, Min Huang, Zheming Gao, and Xingwei Wang. Distributed dynamic pricing of multiple perishable products using multi-agent reinforcement learning. *Expert Systems with Applications*, 237:121252, March 2024.

[108] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 22(268):1–8, January 2021.

[109] Matthew T. Regehr and Alex Ayoub. An Elementary Proof that Q-learning Converges Almost Surely, August 2021.

[110] Patrick Royston. Lowess Smoothing. *Stata Technical Bulletin*, 1(3), January 1992.

[111] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The StarCraft Multi-Agent Challenge, December 2019.

[112] Michael Schlechtinger, Damaris Kosack, Franz Krause, and Heiko Paulheim. By Fair Means or Foul: Quantifying Collusion in a Market Simulation with Deep Reinforcement Learning, June 2024.

[113] Michael Schlechtinger, Damaris Kosack, Heiko Paulheim, and Thomas Fetzer. How algorithms work and play together. *Concurrences : Revue des Droits de la Concurrence*, 2021(3, Article 102088):19–23, 2021.

[114] Michael Schlechtinger, Damaris Kosack, Heiko Paulheim, and Thomas Fetzer. Winning at Any Cost - Infringing the Cartel Prohibition With Reinforcement Learning. In *Advances in Practical Applications of Agents, Multi-Agent Systems, and Social Good. The PAAMS Collection*, Salamanca, Spain, January 2021. Springer Cham.

[115] Michael Schlechtinger, Damaris Kosack, Heiko Paulheim, Thomas Fetzer, and Franz Krause. The Price of Algorithmic Pricing: Investigating Collusion in a Market Simulation with AI Agents. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '23, pages 2748–2750. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, January 2023.

[116] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1889–1897. PMLR, June 2015.

[117] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust Region Policy Optimization, April 2017.

[118] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov, and arXiv:1707.06347. Proximal Policy Optimization Algorithms. https://arxiv.org/pdf/1707.06347, July 2017.

[119] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov, and arXiv:1707.06347. Proximal Policy Optimization Algorithms. https://arxiv.org/pdf/1707.06347, July 2017.

[120] Ulrich Schwalbe. Algorithms, Machine Learning, and Collusion. *Journal of Competition Law & Economics*, 14(4):568–607, January 2018.

[121] Michael Schwind. *Dynamic Pricing and Automated Resource Allocation for Complex Information Services: Reinforcement Learning and Combinatorial Auctions ; with 53 Tables*, volume 589 of *Lecture Notes in Economics and Mathematical Systems*. Springer, Berlin; Heidelberg, January 2007.

[122] Lloyd Shapley and Martin Shubik. Price Strategy Oligopoly with Product Variation*. *Kyklos*, 22(1):30–44, 1969.

[123] Kevin Leyton-Brown Yoav Shoham. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, Cambridge ; New York, illustrated edition edition, February 2009.

[124] Manahan Siallagan, Hiroshi Deguchi, and Manabu Ichikawa. Aspiration-Based Learning in a Cournot Duopoly Model. *Evolutionary and Institutional Economics Review*, 10(2):295–314, January 2013.

[125] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016.

[126] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic Policy Gradient Algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, pages 387–395. PMLR, January 2014.

[127] D. Stahl. Oligopolistic Pricing with Sequential Consumer Search. *The American Economic Review*, 79:700–712, 1989.

[128] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, August 1988.

[129] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, Mass, second edition edition, March 1998.

[130] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction, 2nd Ed.* Reinforcement Learning: An Introduction, 2nd Ed. The MIT Press, Cambridge, MA, US, 2018.

[131] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

[132] Kalyan T. Talluri and Garrett J. van Ryzin. *The Theory and Practice of Revenue Management*. Springer Science & Business Media, February 2006.

[133] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PLOS ONE*, 12(4):e0172395, April 2017.

[134] Gerald Tesauro and Jeffrey O. Kephart. Pricing in Agent Economies Using Multi-Agent Q-Learning. *Autonomous Agents and Multi-Agent Systems*, 5(3):289–304, January 2002.

[135] D. Tjosvold. Cooperation Theory and Organizations. *Human Relations*, 37:743–767, 1984.

[136] A. Turner, Neale Ratzlaff, and Prasad Tadepalli. Avoiding Side Effects in Complex Environments. *ArXiv*, June 2020.

[137] Jon G. Udell. How Important is Pricing in Competitive Strategy? *Journal of Marketing*, 28(1):44–48, January 1964.

[138] United States Department of Justice and Federal Trade Commission. Algorithms and Collusion – Note by the United States, June 2017.

[139] Fumito Uwano, Naoki Tatebe, Yusuke Tajima, Masaya Nakata, Tim Kovacs, and Keiki Takadama. Multi-Agent Cooperation Based on Reinforcement Learning with Internal Reward in Maze Problem. *SICE Journal of Control, Measurement, and System Integration*, 11(4):321–330, July 2018.

[140] Hado van Hasselt, Arthur Guez, and David Silver. Deep Reinforcement Learning with Double Q-Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), March 2016.

[141] Martijn van Otterlo and Marco Wiering. Reinforcement Learning and Markov Decision Processes. In Marco Wiering and Martijn van Otterlo, editors, *Reinforcement Learning: State-of-the-art*, volume v. 12 of *Adaptation, Learning, and Optimization*, pages 3–42. Springer, Heidelberg; New York, January 2012.

[142] John von Neumann, Oskar Morgenstern, and Ariel Rubinstein. *Theory of Games and Economic Behavior (60th Anniversary Commemorative Edition)*. Princeton University Press, 1944.

[143] Ludo Waltman and Uzay Kaymak. Q-learning agents in a Cournot oligopoly model. *Journal of Economic Dynamics and Control*, 32(10):3275–3293, January 2008.

[144] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3):279–292, May 1992.

[145] John Weche and Thomas Weck. Neue Möglichkeiten impliziter Kollusion und die Grenzen des Kartellrechts. *Europäische Zeitschrift für Wirtschaftsrecht*, 1(21):923929, January 2020.

[146] Tobias Werner. *Algorithmic and Human Collusion*. Heinrich Heine University Düsseldorf, January 2021.

[147] Jiawei Xu, Shuxing Li, Rui Yang, Chun Yuan, and Lei Han. Efficient Multi-Goal Reinforcement Learning via Value Consistency Prioritization. *Journal of Artificial Intelligence Research*, 77:355–376, June 2023.

[148] Pei Xu, Junge Zhang, Qiyue Yin, Chao Yu, Yaodong Yang, and Kaiqi Huang. Subspace-aware exploration for sparse-reward multi-agent tasks. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial*

Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence, volume 37 of *AAAI'23/IAAI'23/EAAI'23*, pages 11717–11725. AAAI Press, February 2023.

[149] Pan Ying and Li Dehua. Improvement with Joint Rewards on Multi-agent Cooperative Reinforcement Learning. *2008 International Conference on Computer Science and Software Engineering*, pages 536–539, 2008.

[150] Dan Zhang and Daniel Adelman. An Approximate Dynamic Programming Approach to Network Revenue Management with Customer Choice. *Transportation Science*, 43(3):381–394, August 2009.

[151] Dongxia Zhang, Xiaoqing Han, and Chunyu Deng. Review on the research and practice of deep learning and reinforcement learning in smart grids. *CSEE Journal of Power and Energy Systems*, 4(3):362–370, September 2018.

[152] Tianle Zhang, Zhen Liu, Zhiqiang Pu, and Jianqiang Yi. Peer Incentive Reinforcement Learning for Cooperative Multiagent Games. *IEEE Transactions on Games*, 15(4):623–636, December 2023.

[153] Wen Zhao and Yu-Sheng Zheng. Optimal Dynamic Pricing for Perishable Assets with Nonhomogeneous Demand. *Management Science*, 46(3):375–388, 2000.

[154] Stephan Zheng, Alexander Trott, Sunil Srinivasa, Nikhil Naik, Melvin Gruesbeck, David C. Parkes, and Richard Socher. The AI Economist: Improving Equality and Productivity with AI-Driven Tax Policies, April 2020.

[155] Jinghui Zhong, Xiaomin Hu, Jun Zhang, and Min Gu. Comparison of Performance between Different Selection Strategies on Simple Genetic Algorithms. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA/IAWTIC'06)*, pages 1115–1121, Vienna, Austria, 2005. IEEE.

Appendix

Appendix A

Additional Results for Chapter 3. Investigating Collusion in a Toy Problem

To further challenge and validate the results of the Toy Problem experiment, we conducted additional tests by manipulating the learning rate of the RL algorithms used in the simulations. By varying the learning rates, we aimed to observe how changes in this hyperparameter affect the agents' ability to learn and adapt their strategies over time. This analysis helps in understanding the robustness of the observed collusive behaviors under different learning conditions and provides deeper insights into the dynamics of agent interactions.

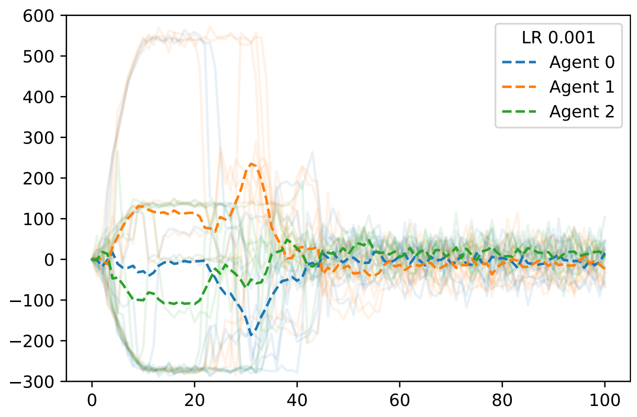


Figure A.1: Learning Rate 0.001.

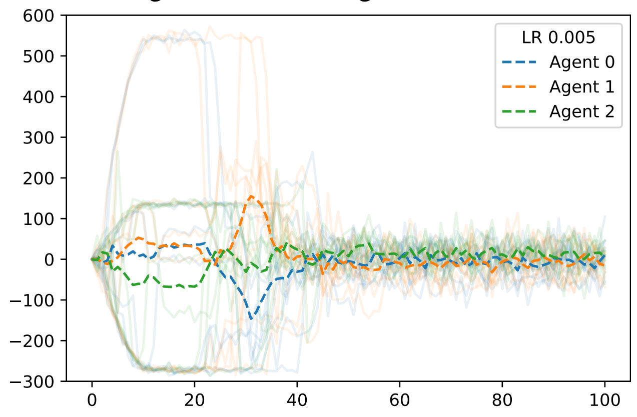


Figure A.2: Learning Rate 0.005.

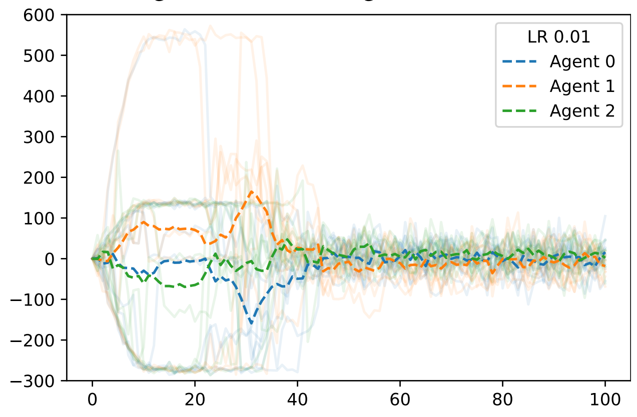


Figure A.3: Learning Rate 0.01.

Figure A.4: Episode reward distribution within the different learning rate scenarios.

Appendix B

Market Environment

B.1 Reinforcement Learning Algorithm Hyperparameter Settings

Table B.1: RL settings and hyperparameter tuning.

Steps per episode	365
Number of training iterations	10000
Hidden Layers	256 x 256
Learning Rate	1e-5
Gradient Clipping (PPO)	1
Batch size (DQN)	250
Batch size (PPO)	4000
Discount Factor γ	1
Replay Buffer Size (DQN)	50000
PPO Clip Param	0.3
Value Function Clipping Param (PPO)	10

B.2 Descriptive statistics of the collusive runs.

Table B.2: Descriptive Statistics of the runs in each scenario.

Scenario	n	μ	Algo	$\overline{P(10000)}$	$\overline{\epsilon(10000)}$	$\sigma\big(P(1000)\big)$	t_{EUC}	Δ
A	3	0	PPO	1.77	9.05	0.01629	1060	1.0749
			DQN	1.75	8.91	0.01799	5247	1.07001
		0.5	PPO	1.87	7.48	0.0644	2532	0.88613
			DQN	1.78	6.03	0.00337	/	/
		1	PPO	2.0	5.57	0.12697	/	/
			DQN	2.0	4.85	0.13228	/	/
	5	0	PPO	1.75	6.15	0.03123	1282	1.22834
			DQN	1.69	5.61	0.02267	6223	1.12777
		0.5	PPO	1.83	4.34	0.03149	797	0.84309
			DQN	1.91	4.13	0.06675	/	/
		1	PPO	1.95	3.11	0.0641	8004	0.63864
			DQN	2.17	2.9	0.0947	/	/
B	3	0	PPO	1.76	7.8	0.01961	958	0.94034
		0.5	PPO	1.73	6.54	0.02549	1236	0.80212
		1	PPO	1.75	5.1	0.03248	1049	0.61831
	5	0	PPO	1.74	5.64	0.02323	907	1.12276
		0.5	PPO	1.75	4.15	0.02231	530	0.84312
		1	PPO	1.82	2.92	0.02335	722	0.57719

B.3 Heatmaps for Market-Price Combinatorics

In this section, we present a series of heatmaps to explore the combinatorial price strategies of agents in our economic environment. To rigorously test the market conditions, we disable any stochasticity in subjective consumer behavior and set μ to 0, thereby ensuring that consumers select products based on the roulette wheel selection strategy. The primary objective of this analysis is to determine which price combinations Agent 0 and Agent 2 can choose to maximize their revenue when Agent 1 selects specific prices to achieve maximum revenue. To provide this insight, the following heatmaps illuminate the joint revenue outcomes for various price combinations. We observe that the simulation settings incentivize agents to overbid or underbid their competitors' prices strategically. This strategy exploits consumers' willingness to purchase products at slightly higher prices, which is facilitated by setting $\mu = 0$.

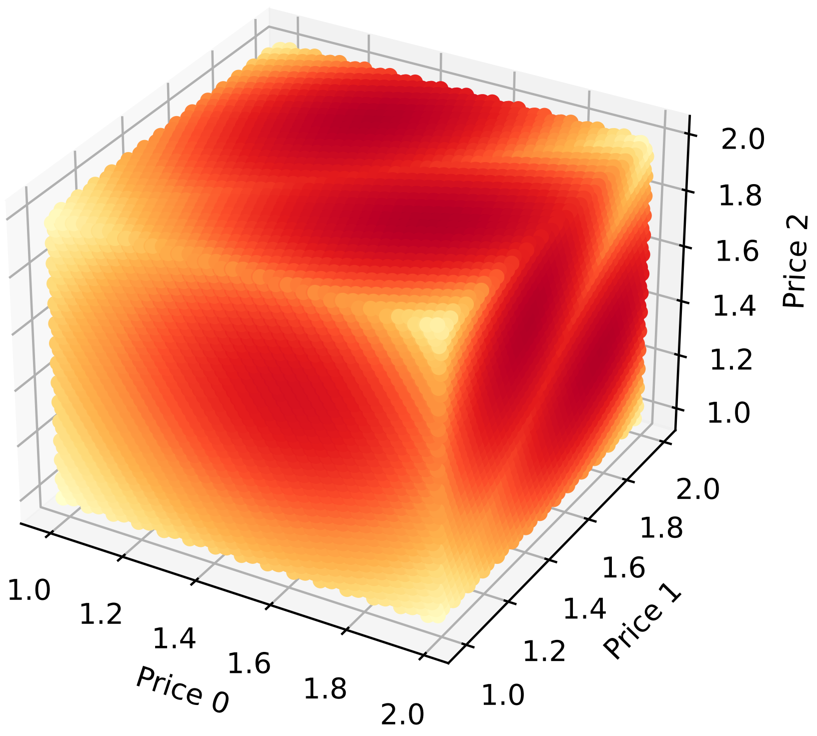
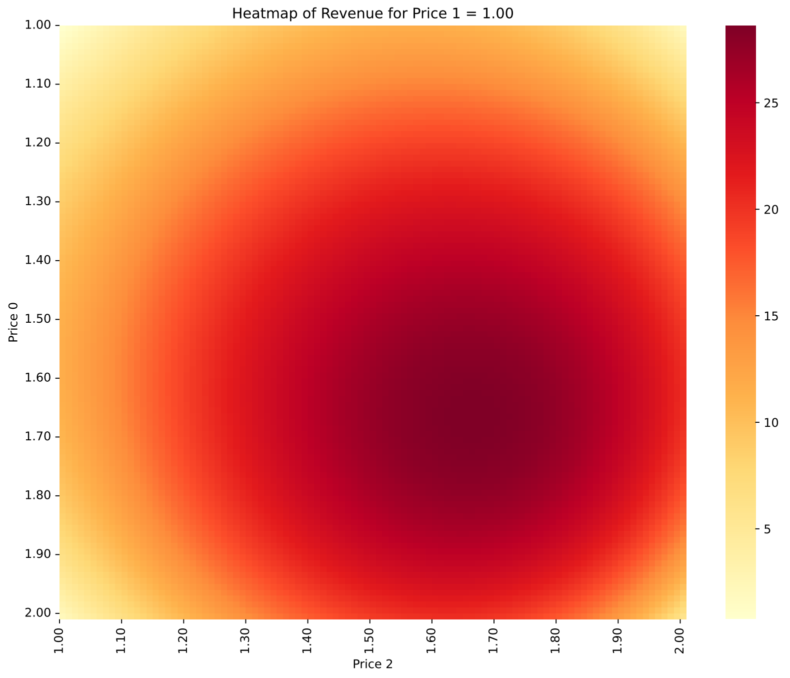


Figure B.1: 3D-heatmap visualization of the possible combinatorics and revenues when three agents set prices.



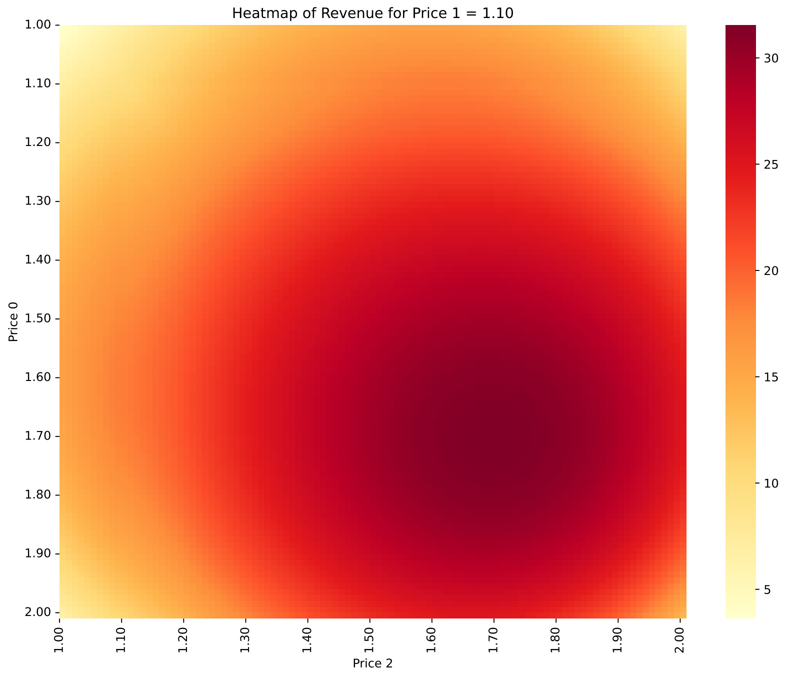
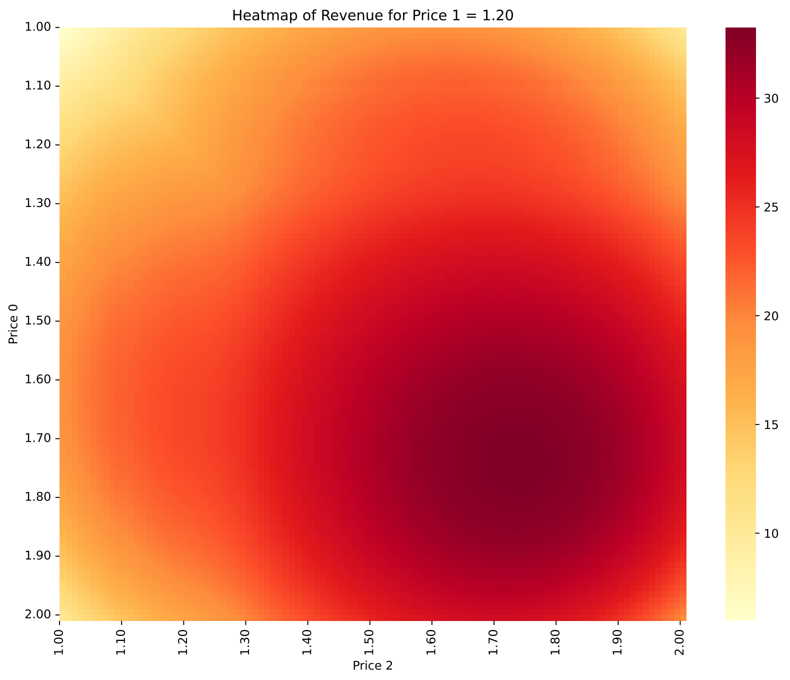


Figure B.2: Heatmap of possible joint revenues, when agent 1 sets the prices 1.00 and 1.10.



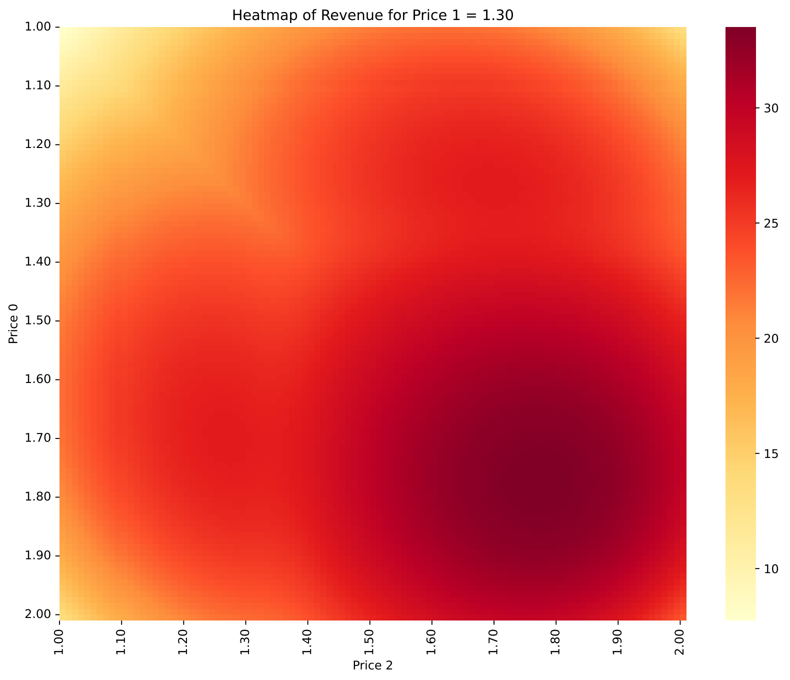
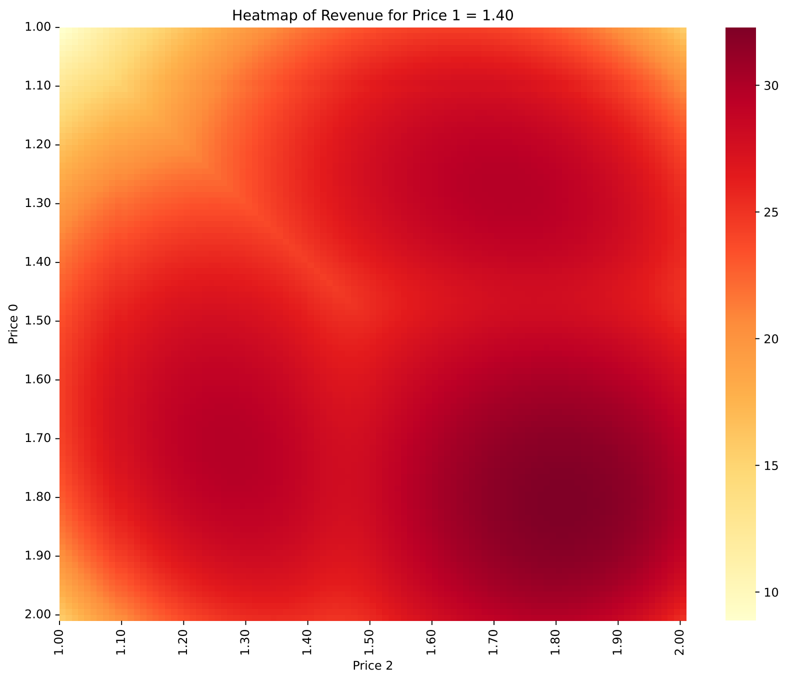


Figure B.3: Heatmap of possible joint revenues, when agent 1 sets the prices 1.20 and 1.30.



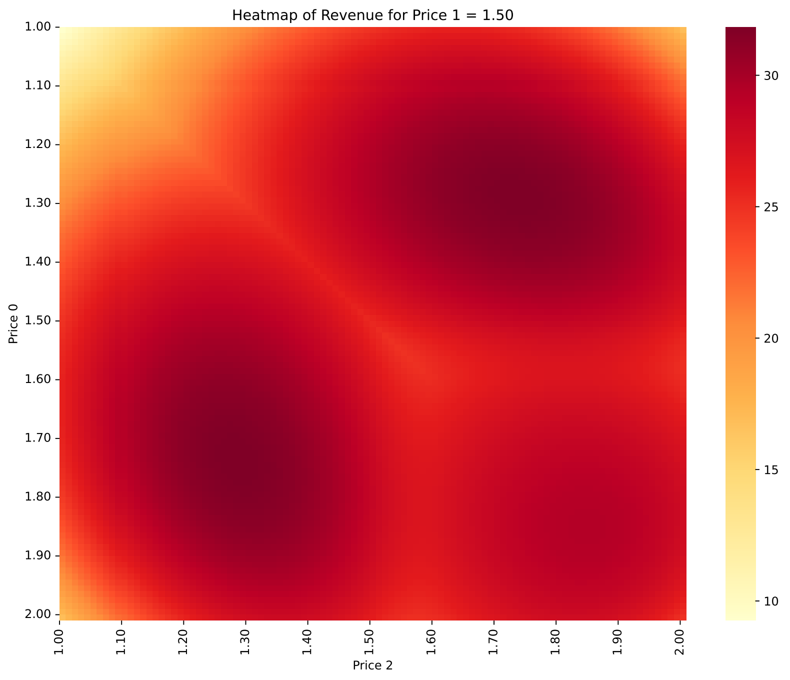
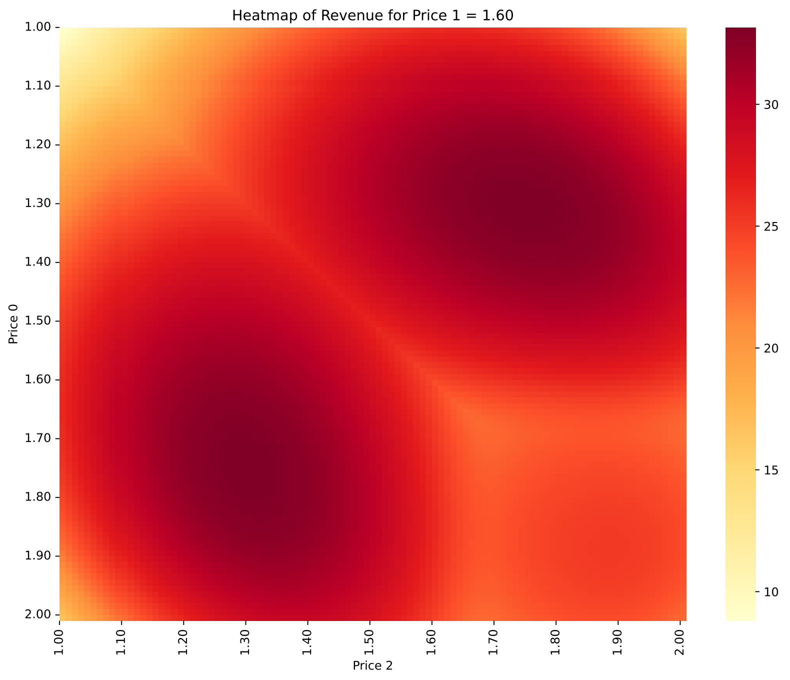


Figure B.4: Heatmap of possible joint revenues, when agent 1 sets the prices 1.40 and 1.50.



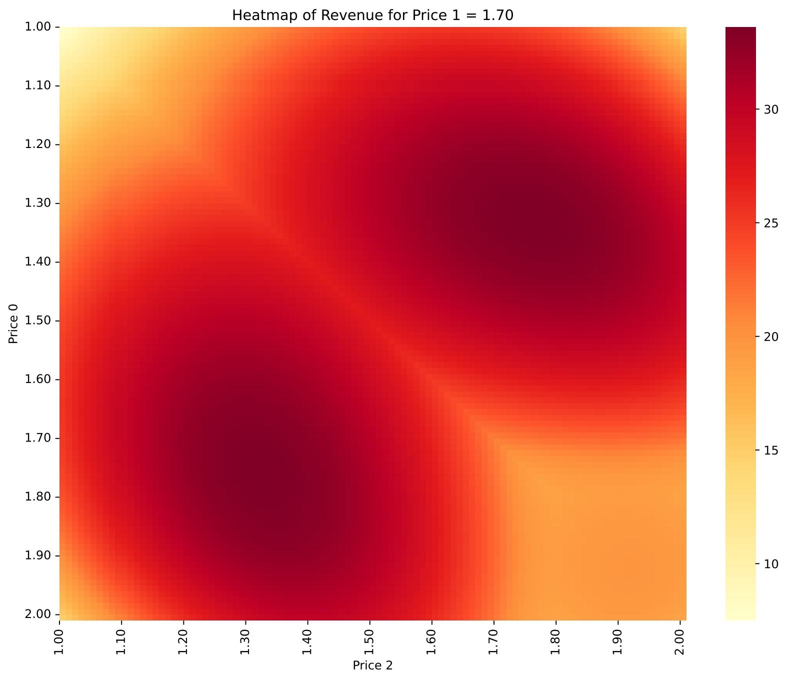
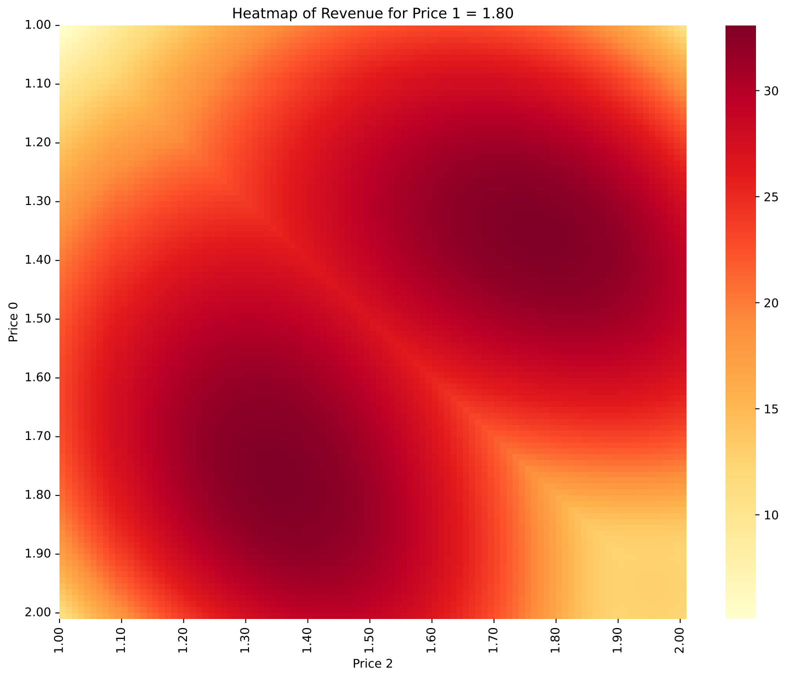


Figure B.5: Heatmap of possible joint revenues, when agent 1 sets the prices 1.60 and 1.70.



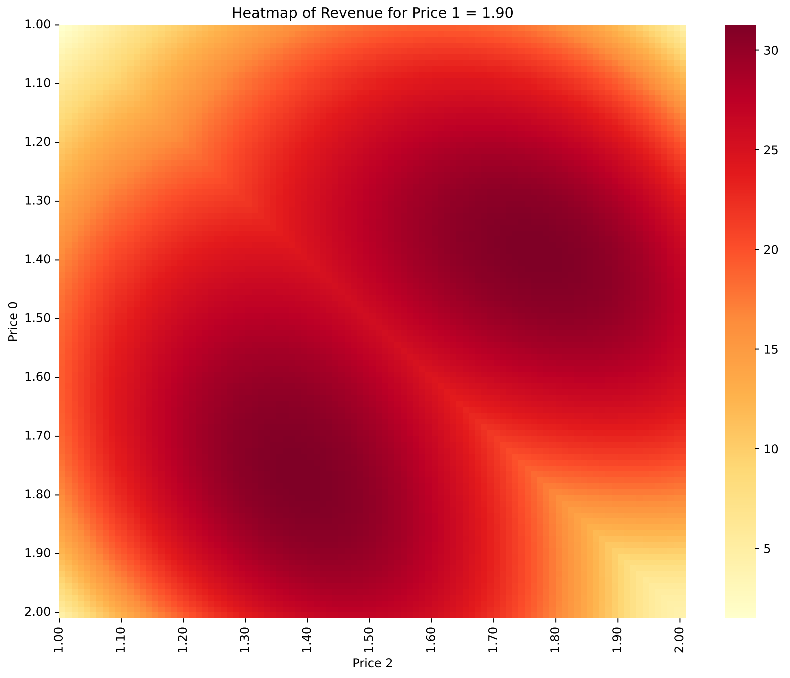


Figure B.6: Heatmap of possible joint revenues, when agent 1 sets the prices 1.80 and 1.90.

Appendix C

Ablation Study

In addition to the main scenarios of this study, we aim to understand the importance of certain simulation and algorithm (hyper-)parameters and how they collectively affect the agents' behavior. Similar to the main scenarios, the prices and profits of the episodes are computed as the mean values derived from the corresponding step prices, which have also undergone LOWESS smoothing. Each experimental iteration comprises 10,000 episodes, each encompassing 365 sequential steps. Every plot comprises the price and profit achieved in the episodes and an insight into the individual step pricing of the last episode. We employ PPO during these runs.

C.1 Altering the Number of Agents

Initially, we endeavor to scrutinize the influence exerted by both the quantity of agents and their respective market entry timings upon the outcomes derived from the simulation. To execute that, we simulated with 15 agents as well as two simulations where agents join throughout the run.

C.2 Utilizing 15 Agents (Figures C.1, C.2, C.3)

We discern a pricing behavior that mirrors the outcomes seen in the scenarios featuring fewer agents. Nevertheless, the visually intricate and seemingly stochastic pricing trajectory contrasts distinctly with the notably coherent profit graph. Adding to that, we observe the same oscillation pattern within the step pricing behavior.

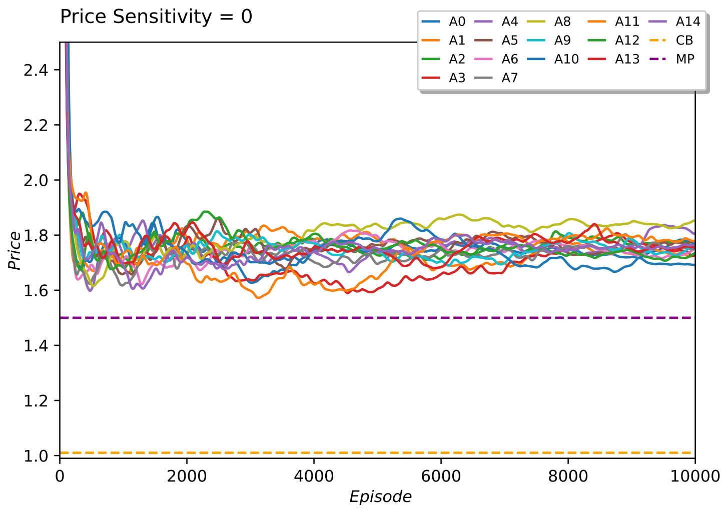


Figure C.1: PPO, 15 Agents (Part 1).

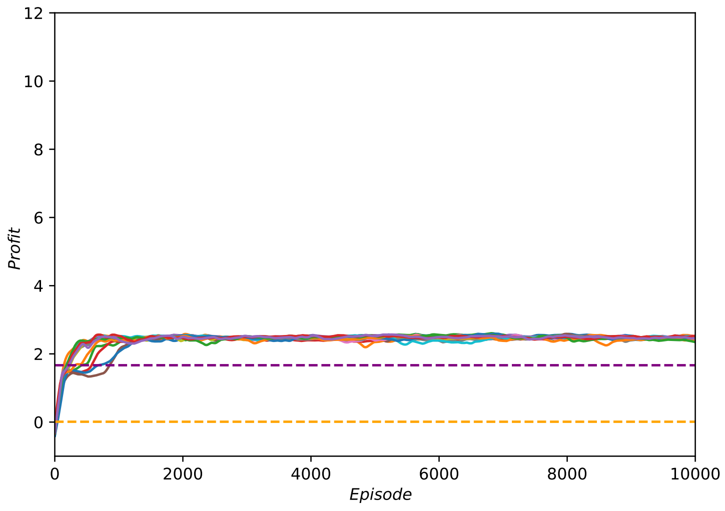


Figure C.2: PPO, 15 Agents (Part 2).

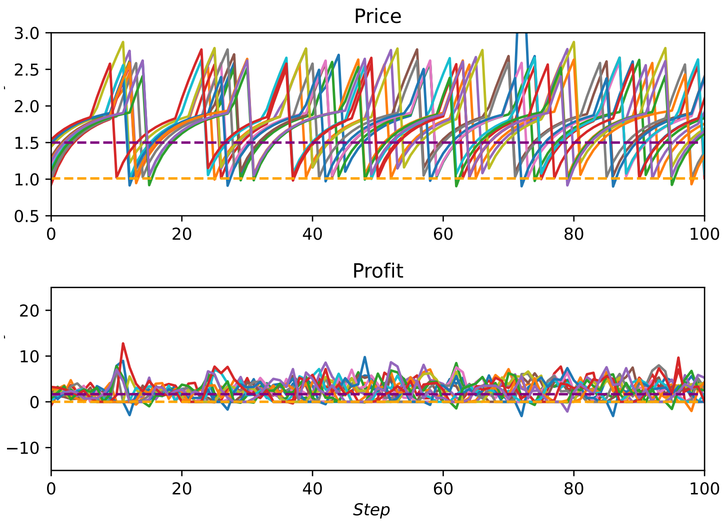


Figure C.3: PPO, 15 Agents (Part 3).

C.3 Repercussions of Agents that Join Later

To conduct a more comprehensive analysis of the influence wielded by varying the number of agents, we employ a run starting with two agents. In episodes 3000 and 6000 an additional agent is introduced to the market. We aim to investigate the response of pre-trained agents to this alteration. The primary run (cf. Figures C.4, C.5, C.6) adheres to conventional settings akin to scenario A, while the secondary run (cf. Figures C.7, C.8, C.9) modifies the agents' observation space by concealing competitors' pricing information.

The insertion of a new agent prompts a behavioral recalibration among the existing agents. Following a noticeable disruption within the graphs, this leads to a gradual collective adjustment of the already trained agents towards a reduced profit level, while pricing remains relatively stable. Notably, the main difference between the two scenarios lies in the repercussions of the untrained agent's market entry. In scenario A, discernible disruptions manifest in both pricing strategies and profit dynamics. Conversely, scenario B demonstrates a comparably smoother adaptation process. We attribute this behavior to the reduced amount of information necessary to solve the problem. The step-wise pricing behavior and chosen strategies remain consistent with the main approach of this paper.

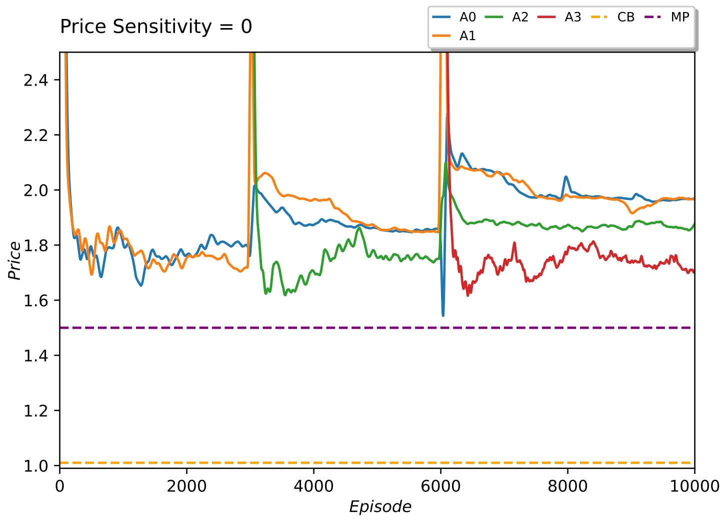


Figure C.4: PPO, 3 Agents, One Agent Joining at Episode 3000, One Agent Joining at Episode 6000 (Part 1).

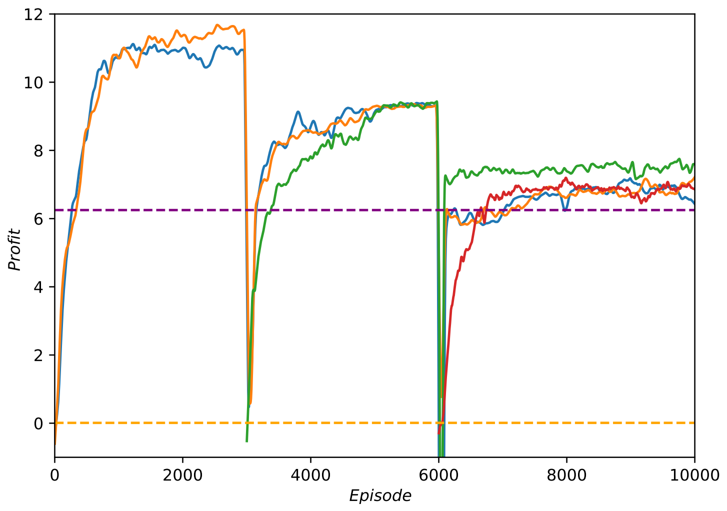


Figure C.5: PPO, 3 Agents, One Agent Joining at Episode 3000, One Agent Joining at Episode 6000 (Part 2).

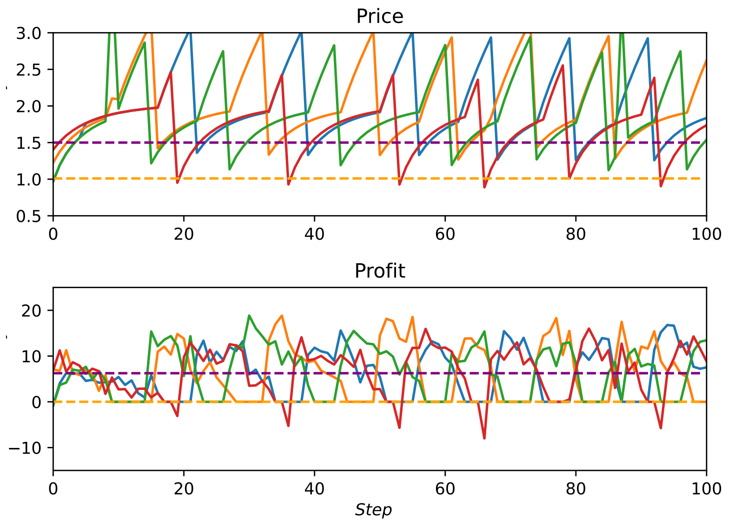


Figure C.6: PPO, 3 Agents, One Agent Joining at Episode 3000, One Agent Joining at Episode 6000 (Part 3).

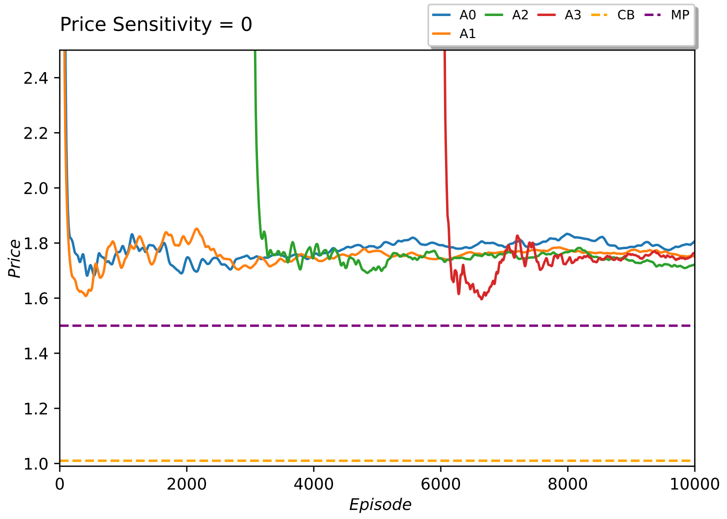


Figure C.7: PPO, Blind, 3 Agents, One Agent Joining at Episode 3000, One Agent Joining at Episode 6000 (Part 1).

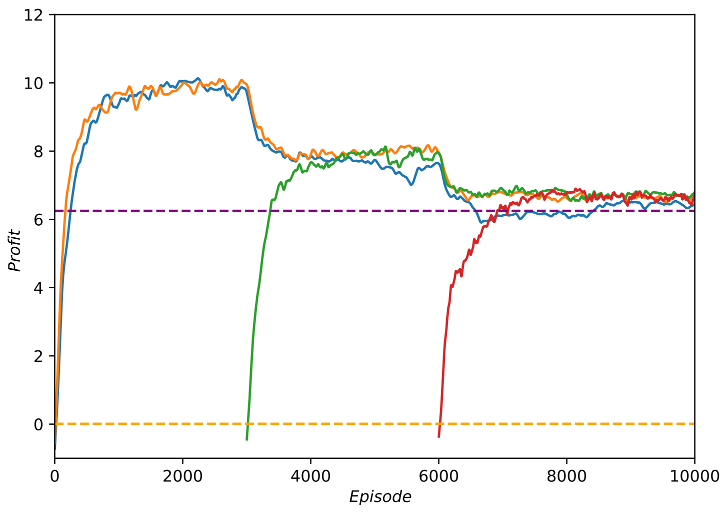


Figure C.8: PPO, Blind, 3 Agents, One Agent Joining at Episode 3000, One Agent Joining at Episode 6000 (Part 2).

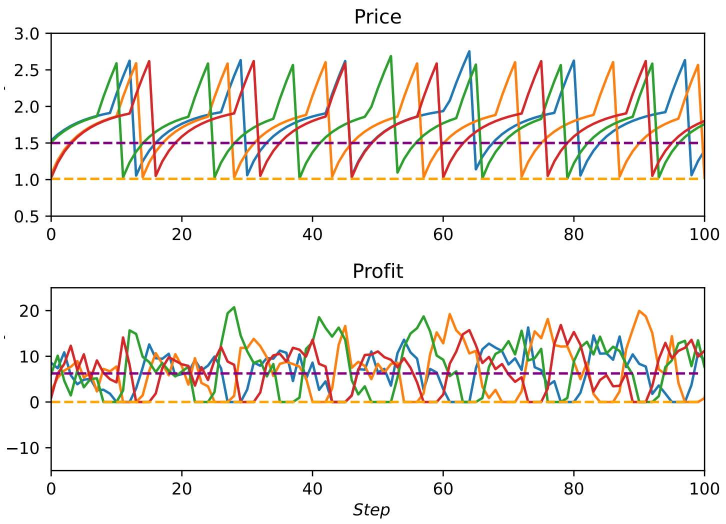


Figure C.9: PPO, Blind, 3 Agents, One Agent Joining at Episode 3000, One Agent Joining at Episode 6000 (Part 3).

C.4 Altering the Learning Rate

Subsequent to this point, each experimental iteration incorporates 'blind' agents, characterized by a constrained action space, with the intention of potentially heightening collaboration challenges. In this context, we adjusted the agents' learning rates from 1e-5 to 1e-4 and 5e-5. This modification aims to highlight potential variations in learning speed in order to impede the agents' capacity to establish a collusive outcome. The results are highlighted in Figures C.10, C.11, C.12, C.13, C.14, and C.15. The modifications to the learning rates do not result in visible impacts on agent performance, selected strategies, or discernible patterns. Nevertheless, we observe a divergence within the later stages of the run associated with a learning rate of 5e-5. Upon attaining an equilibrium-like state, agent 0 enacts a reduction in its pricing strategy, thereby yielding a diminished profit margin. Given the predominance of runs converging toward equilibrium within a certain episode range, we anticipate this isolated occurrence to be an outlier. Nonetheless, this divergence could be a potential lead toward collusion prevention.

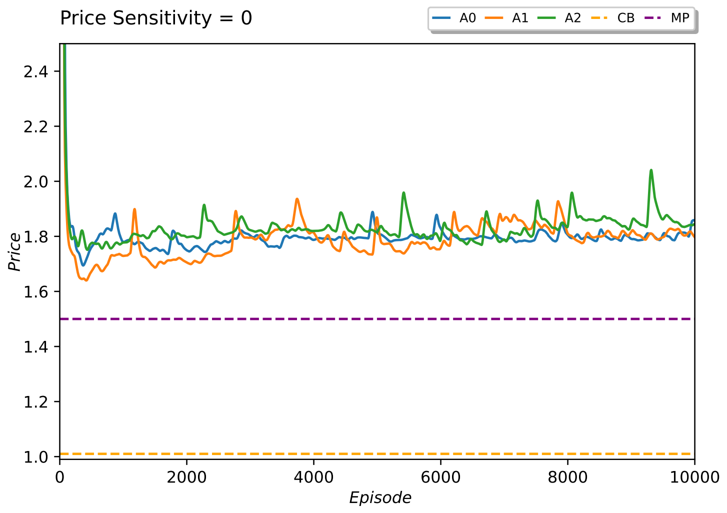


Figure C.10: PPO, Blind, 3 Agents, Learning Rate: 1e-4 (Part 1).

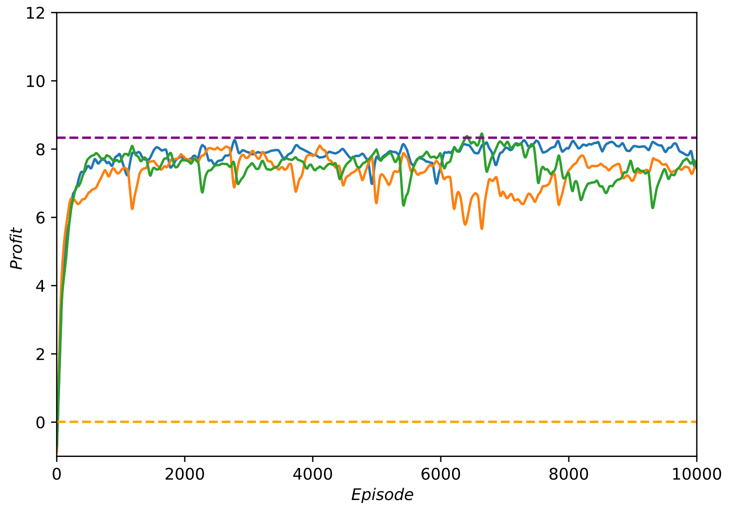


Figure C.11: PPO, Blind, 3 Agents, Learning Rate: 1e-4 (Part 2).

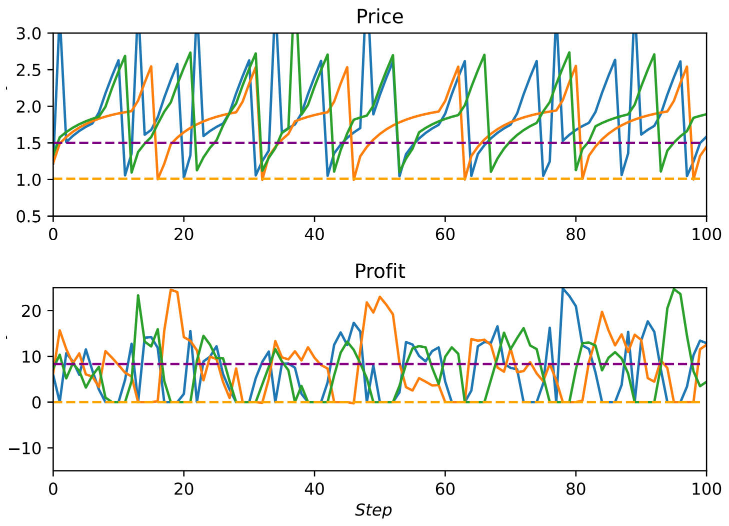


Figure C.12: PPO, Blind, 3 Agents, Learning Rate: 1e-4 (Part 3).

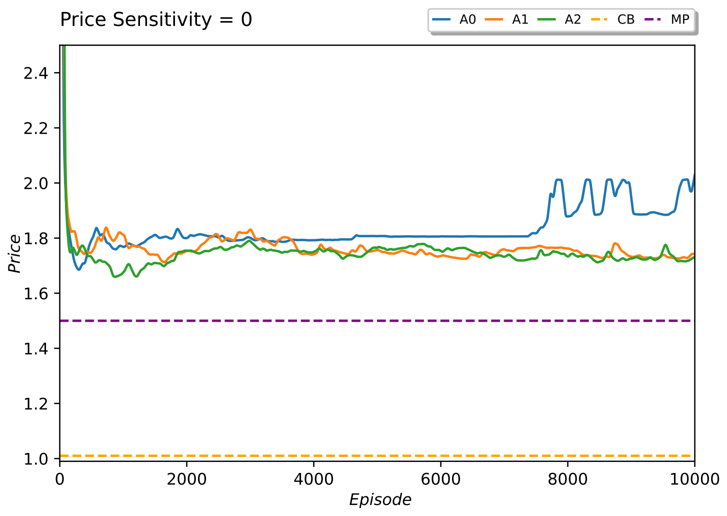


Figure C.13: PPO, Blind, 3 Agents, Learning Rate: 5e-5 (Part 1).

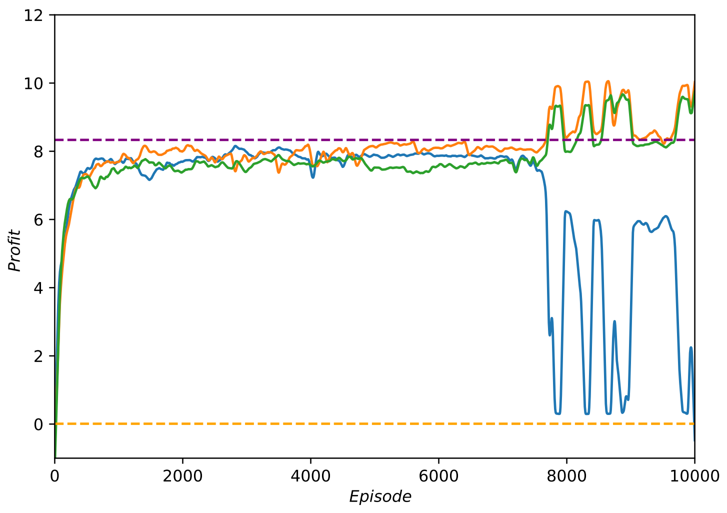


Figure C.14: PPO, Blind, 3 Agents, Learning Rate: 5e-5 (Part 2).

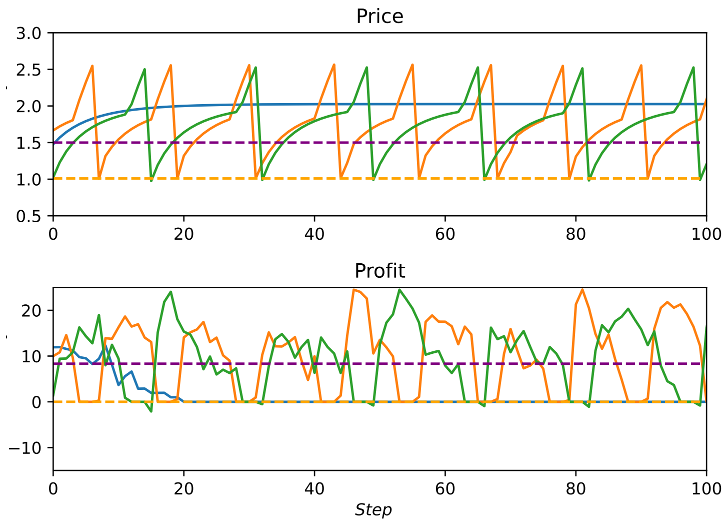


Figure C.15: PPO, Blind, 3 Agents, Learning Rate: 5e-5 (Part 3).

C.5 Altering the Hidden Layer Neurons

We reduced the neurons for the two hidden layers from 256x256 to 128x128 and 64x64 respectively. With this simplification of the neural networks, we aim to impede the agents' ability to achieve a collusive outcome. The results are highlighted in Figures C.16, C.17, C.18, C.19, C.20, and C.21. Despite this significant modification, the agents were able to achieve and sustain a collusive outcome.

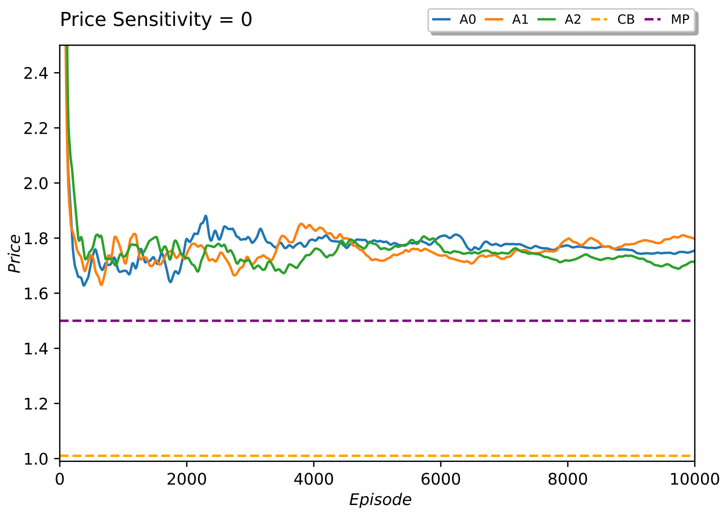


Figure C.16: PPO, Blind, 3 Agents, Hidden Layer Neurons: 128x128 (Part 1).

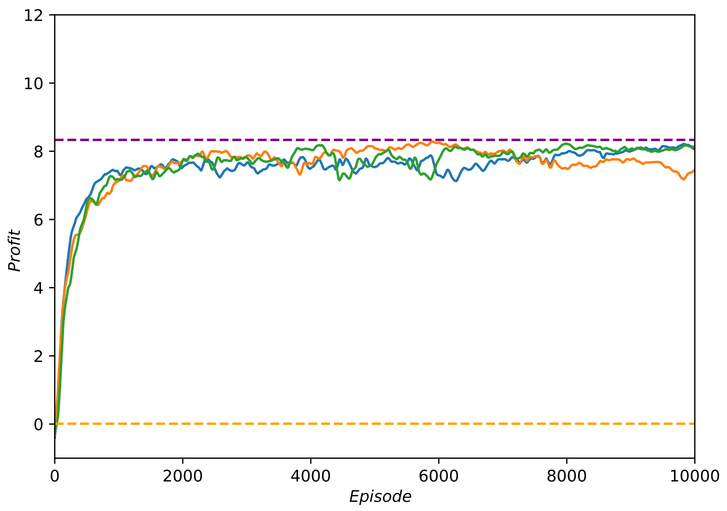


Figure C.17: PPO, Blind, 3 Agents, Hidden Layer Neurons: 128x128 (Part 2).

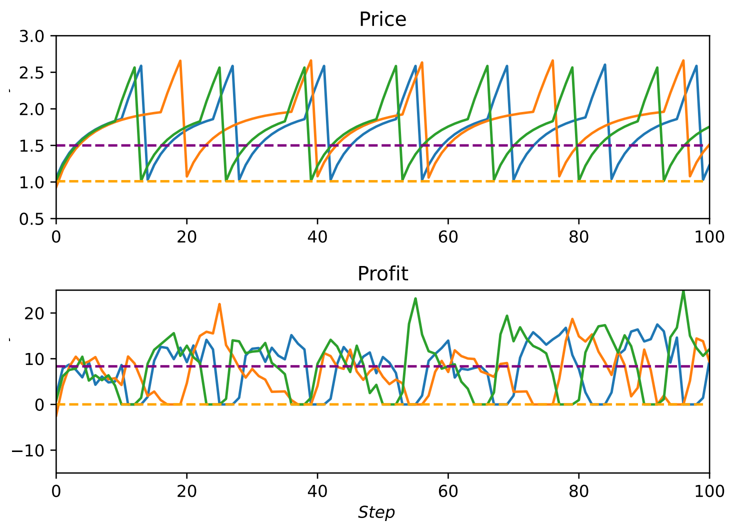


Figure C.18: PPO, Blind, 3 Agents, Hidden Layer Neurons: 128x128 (Part 3).

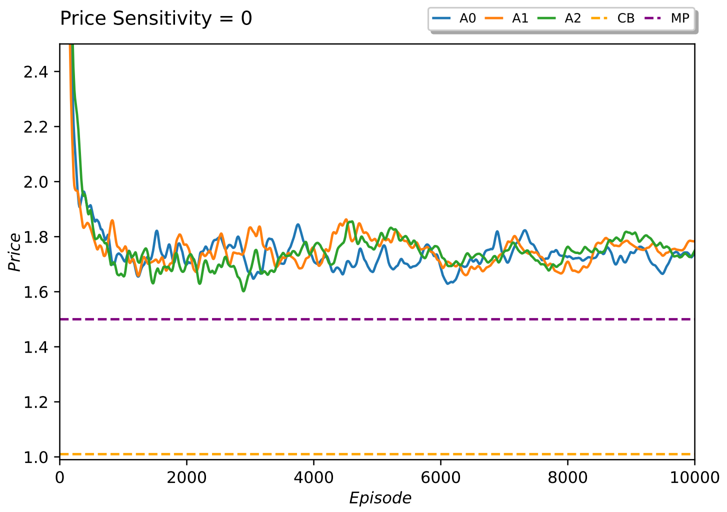


Figure C.19: PPO, Blind, 3 Agents, Hidden Layer Neurons: 64x64 (Part 1).

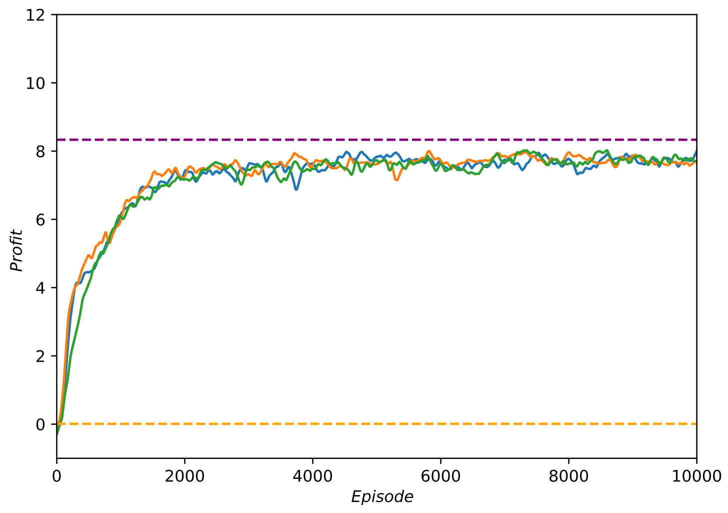


Figure C.20: PPO, Blind, 3 Agents, Hidden Layer Neurons: 64x64 (Part 2).

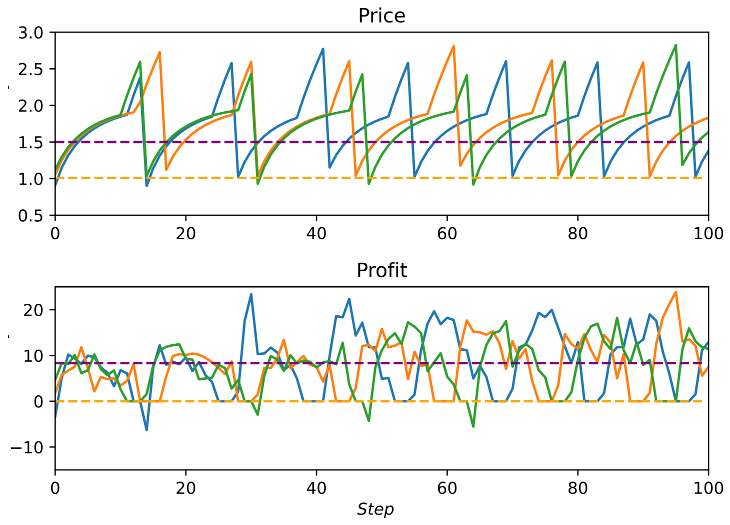


Figure C.21: PPO, Blind, 3 Agents, Hidden Layer Neurons: 64x64 (Part 3).

C.6 Altering the Action Gradients

In order to prompt different strategies, we increased the action gradients from 7 to 21, 51, and 71 respectively. The results are highlighted in Figures C.22, C.23,C.24, C.25, C.26, C.27, C.28, C.29, and C.30. This adjustment allows the agents to finely calibrate pricing, albeit at the expense of heightened computational intricacy. This adjustment generally leads to an increased profit with similar prices, a more severe price stagnation, as well as a quicker time to converge. However, the strategies applied by the agents significantly diverge from the ones applied in the main scenarios of this study.

Employing 21 gradient steps yields the most substantial profit accumulation observed across the experimental sessions, surpassing even the profits attained within scenario A. The agents accomplish this feat through an approach that exhibits distinct characteristics while maintaining a fundamental similarity. Each agent follows its own unique oscillation pattern, yet collectively, they converge toward comparable long-term profit levels. Analogous to the main scenarios of this paper, the agents adeptly capitalize on the simulation dynamics by designating a single agent to sell at a reduced price, thereby enabling the remaining agents to price above the monopolistic threshold. We witness analogous behavior when applying 51 and 71 gradients. Despite the heightened array of action choices resulting from the increased gradients, the agents converge toward a state of market stasis characterized by recurrent price repetition. Nevertheless, within this state of repetition, one of the agents persists in executing an oscillation pattern, thereby propelling the collective profit beyond the monopolistic benchmark.

This adjustment not only shows that there seems to be a sweet spot for the agents (ca. 21 gradients), but it also highlights that, with proper parametrization, sellers could potentially achieve higher profits as well as an equilibrium state that supersedes the ”unaltered” scenarios despite the restricted observation space.

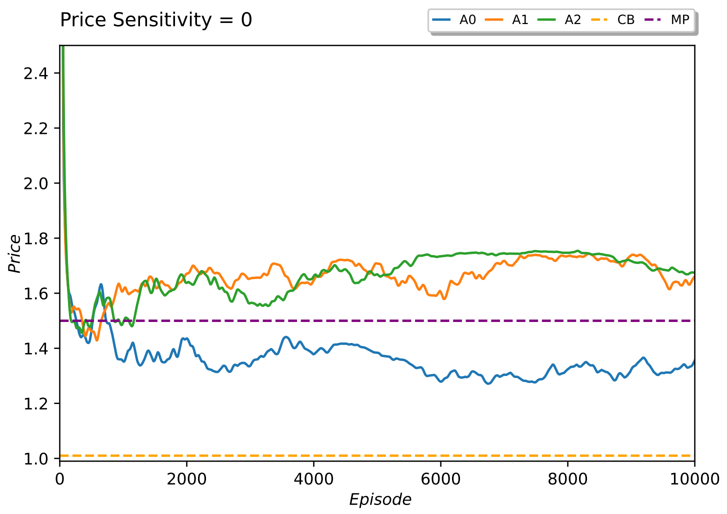


Figure C.22: PPO, Blind, 3 Agents, 21 Action Gradients (Part 1).

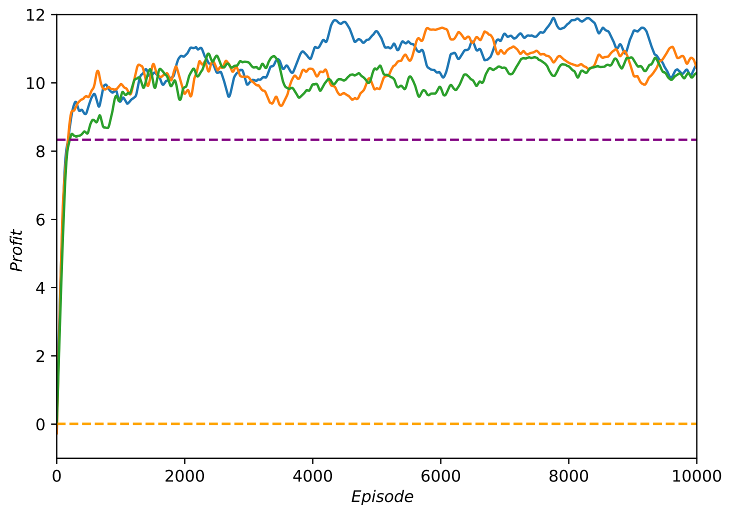


Figure C.23: PPO, Blind, 3 Agents, 21 Action Gradients (Part 2).

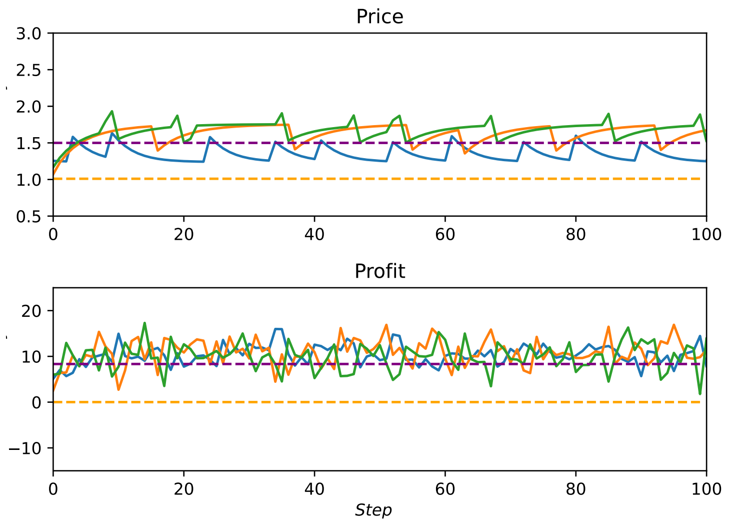


Figure C.24: PPO, Blind, 3 Agents, 21 Action Gradients (Part 3).

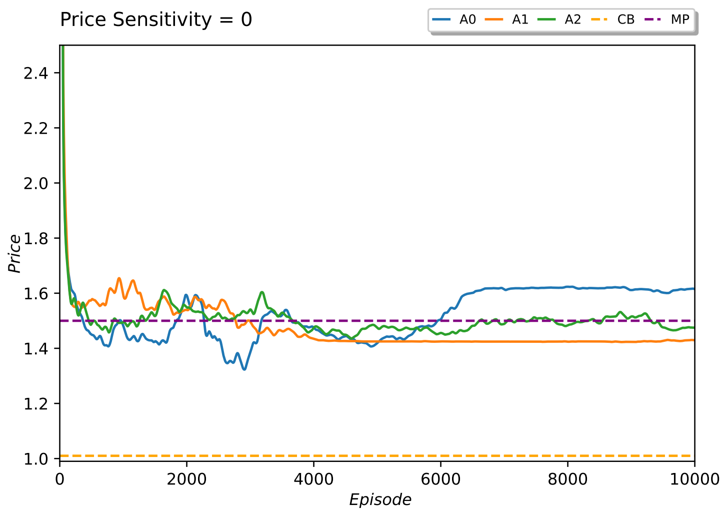


Figure C.25: PPO, Blind, 3 Agents, 51 Action Gradients (Part 1).

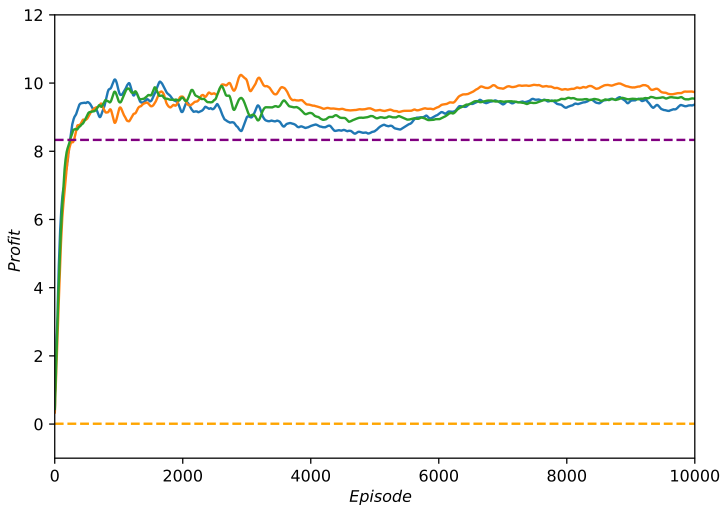


Figure C.26: PPO, Blind, 3 Agents, 51 Action Gradients (Part 2).

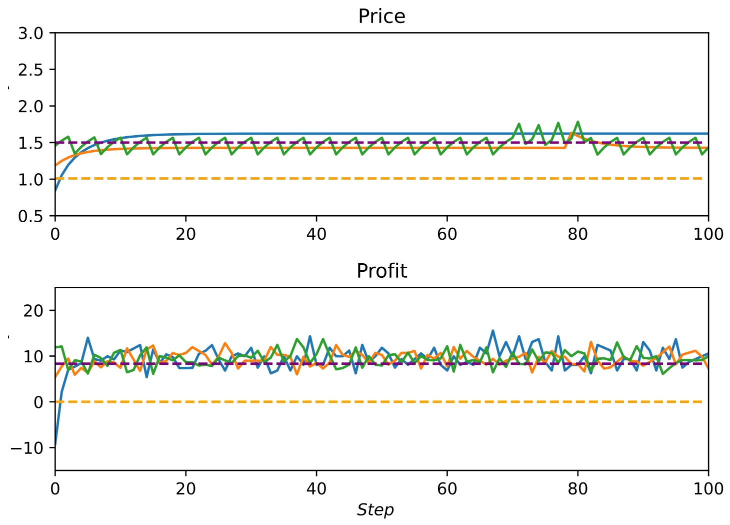


Figure C.27: PPO, Blind, 3 Agents, 51 Action Gradients (Part 3).

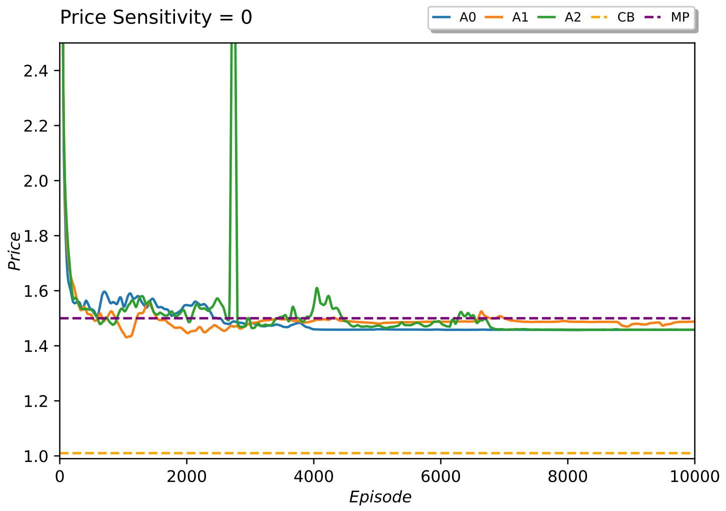


Figure C.28: PPO, Blind, 3 Agents, 71 Action Gradients (Part 1).

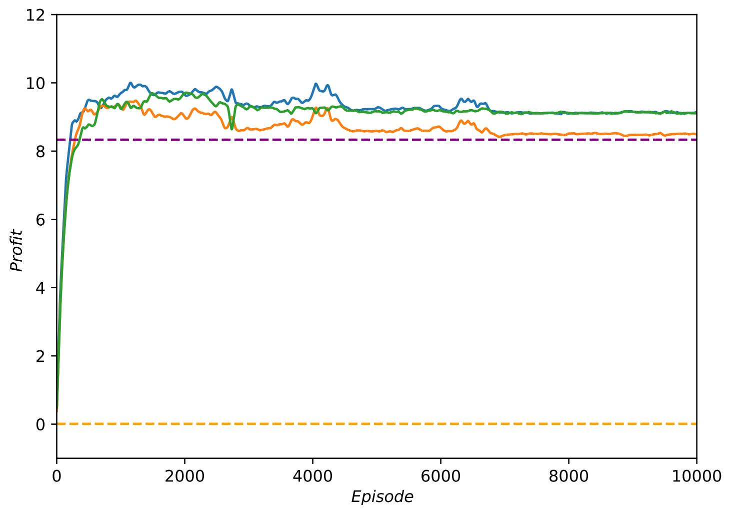


Figure C.29: PPO, Blind, 3 Agents, 71 Action Gradients (Part 2).

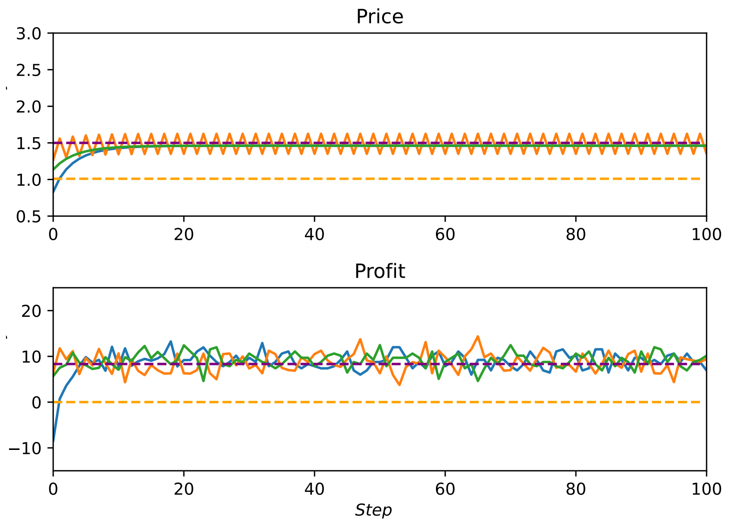


Figure C.30: PPO, Blind, 3 Agents, 71 Action Gradients (Part 3).

Appendix D

Analysis Settings

D.1 Configuration of DecisionTreeClassifier

Below is a table detailing the configuration parameters used for the `DecisionTreeClassifier` from the `sklearn.tree` module.

Parameter	**Default Value**	**Description**
Criterion	`gini`	The function to measure the quality of a split. 'Gini' measures Gini impurity.
Splitter	`best`	Strategy to choose the split at each node. 'Best' selects the best split.
Max Depth	`None`	The maximum depth of the tree. Nodes are expanded until all leaves are pure or until all leaves contain less than `min_samples_split` samples.
Min Samples Split	2	The minimum number of samples required to split an internal node.
Min Samples Leaf	1	The minimum number of samples required to be at a leaf node.
Max Features	`None`	The number of features to consider when looking for the best split. Considering all features when not set.

Table D.1: Configuration parameters for DecisionTreeClassifier.

D.2 Configuration of DecisionTreeRegressor

Below is a table detailing the configuration parameters used for the `DecisionTreeRegressor` from the `sklearn.tree` module.

Parameter	**Default Value**	**Description**
Criterion	`mse`	Measures the quality of a split using mean squared error, which captures the average of the squares of the errors.
Splitter	`best`	Strategy to choose the split at each node. 'Best' selects the best split according to the criterion.
Max Depth	`None`	The maximum depth of the tree, expanding until all leaves are pure or until leaves contain fewer than `min_samples_split` samples.
Min Samples Split	2	The minimum number of samples required to split an internal node.
Min Samples Leaf	1	The minimum number of samples required to be at a leaf node.
Max Features	`None`	The number of features to consider when looking for the best split, considering all features if not set.
Max Leaf Nodes	`None`	The maximum number of leaf nodes allowed. No limit if not set.
Min Impurity Decrease	0	A node will be split if this split induces a decrease of impurity greater than or equal to this value.

Table D.2: Configuration parameters for DecisionTreeRegressor.

Appendix E

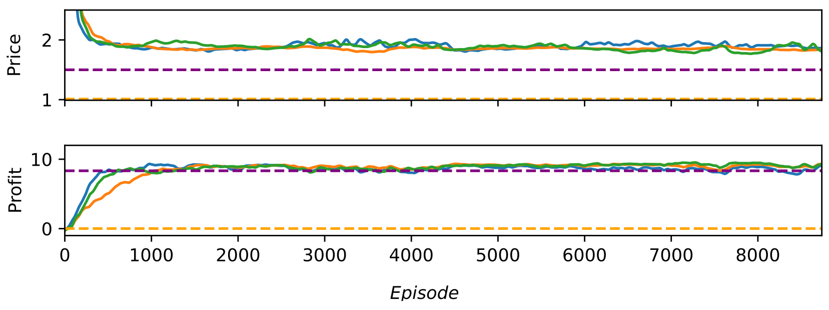
Additional Results For Chapter 6. Preventing Collusion

Chapter 6 highlights results from both the initial, unsuccessful runs and the final, successful runs. This appendix aims to fill the gaps of this iterative process. To make our thought process more transparent, we present a selected amount of experimental results in each of the following sections. While we applied both classification and regression in our attempts to deter the agents from colluding, we will only present the classification results here to avoid inflating the page count. An example of a regression result can be found in Chapter 6.

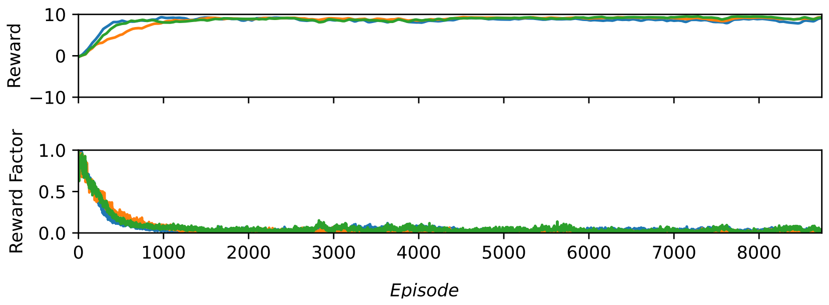
E.1 Supervision V2

The following runs were performed using dense punishment in combination with the profit achieved by the agents. As we were unable to deter the agents from playing collusively, we aimed to reward anti-cartel behavior by calculating the new reward using $r_t^i * (1 + r_s(i, t))$.

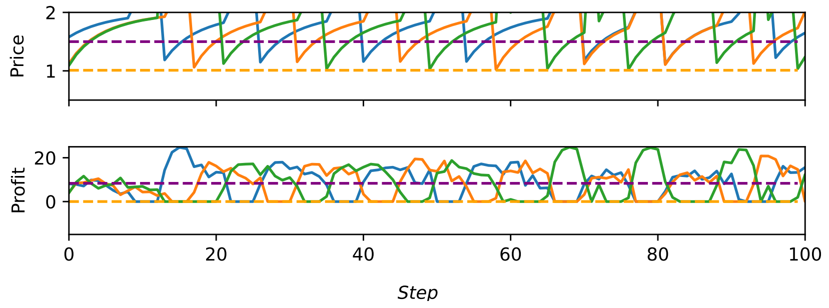
The results shown in Figure E.1 indicate that the agents were unable to effectively learn to balance both reward signals and instead preferred to pursue a stable, collusive outcome.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor



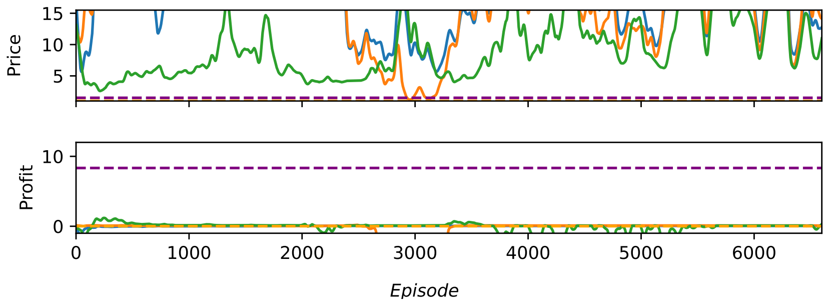
(c) Stepwise Price and Profit

Figure E.1: Dense supervision reward-based run, that aims to reward anti-cartel behavior, measured using classification methods.

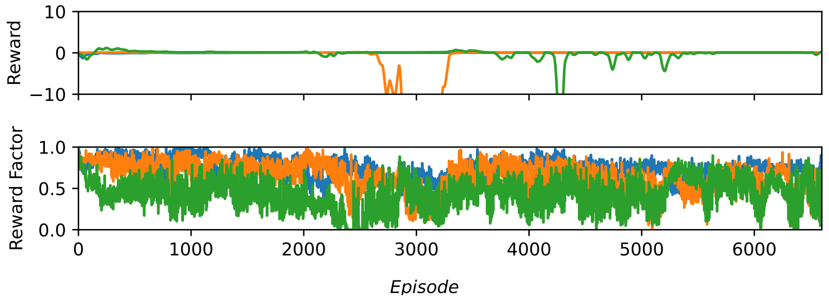
E.2 Supervision V3

As we were unable to deter the agents from achieving a collusive outcome, we investigated the learning behavior when the agents were trained to play unpredictably, thereby maximizing $r_s(i, t)$.

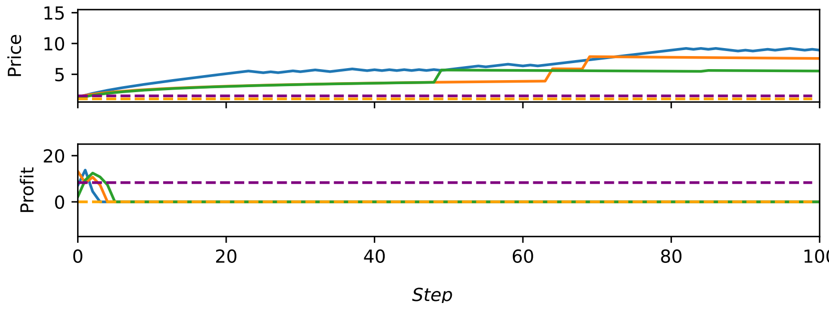
The results (cf. Figure E.2) show that the agents can achieve a high supervision reward and thus learn to play unpredictably throughout a run. However, since profit-oriented behavior was not rewarded, we observed random pricing actions leading to low sales.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor



(c) Stepwise Price and Profit

Figure E.2: Only rewarding the agents using supervision metrics and no profit-related reward.

E.3 Supervision V4

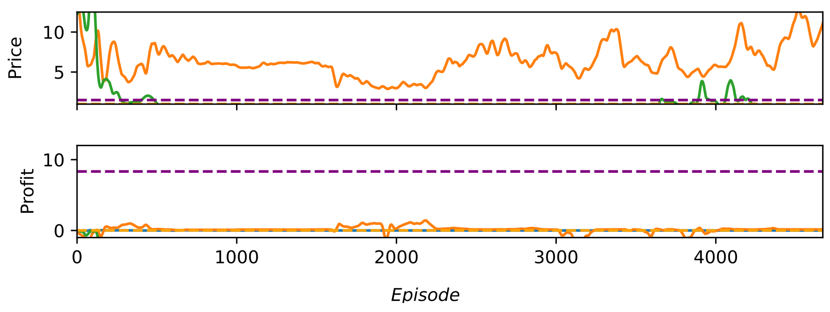
The following experiments incorporate time series analysis factors into the supervision reward calculations. As explained in Chapter E, we include autocorrelation, volatility (standard deviation), and entropy. These methods aim to provide agents with further insights into their learning objectives and deter them from achieving collusive outcomes. In the following runs, the agents were rewarded solely based on the supervision reward derived from the time series analysis, as well as mixed with the profit-based reward. Unlike metrics such as classification accuracy or MAPE in regression, these metrics do not have a clearly defined numerical ceiling. Therefore, we allowed the agents to determine how to maximize the reward, rather than restricting them to a value between 0 and 1.

The results of the first experiment (cf. Figure E.3) show that the agents successfully understood how to increase the reward factor. However, similar to using only classification or regression as a reward, the agents were not able to make profit-driven decisions. Combining the time series reward with the profit-related reward, as in V2 ($r_t^i * (1 + r_s(i,t))$), yielded slightly different results (cf. Figure E.4). This combination led to a situation where agents chose to maximize only one of the reward components. In this run, the agents opted to maximize the time series-based supervision reward while attempting to achieve a profit-related reward of zero. This resulted in price-setting behavior outside the feasible boundaries of the given market simulation.

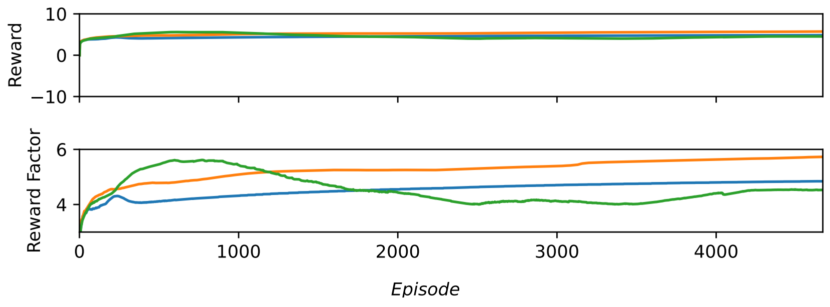
Observing that agents exploited the time series reward mix by selecting prices between 0 and 1, we adjusted the reward calculation to differently handle cases where r_t^i is negative:

$$f(r_t^i, r_s(i,t)) = \begin{cases} r_t^i \cdot r_s(i,t) & \text{if } r_t^i \geq 0 \\ \frac{r_t^i}{r_s(i,t)} & \text{if } r_t^i < 0 \end{cases} \tag{E.1}$$

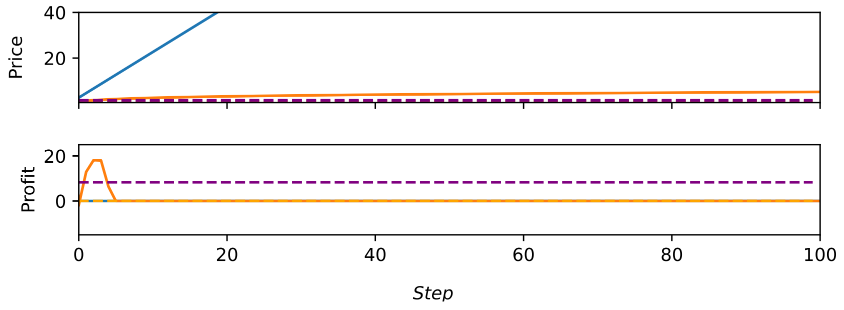
However, this adjustment did not result in different behavior.



(a) Price and Profit

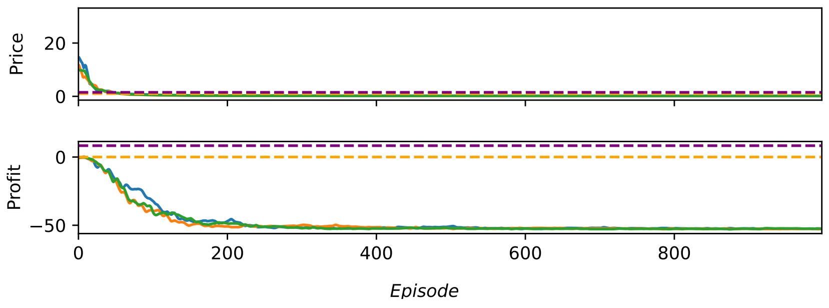


(b) Rewards and Supervision Reward Factor

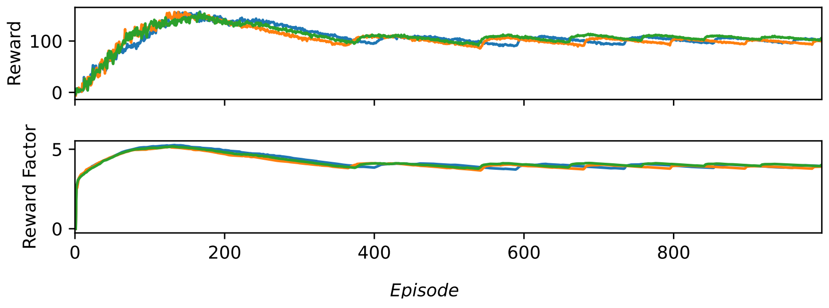


(c) Stepwise Price and Profit

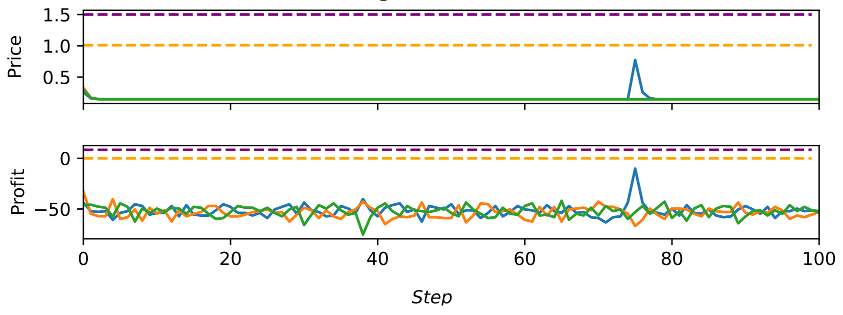
Figure E.3: Training only using the time series-based supervision reward.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor



(c) Stepwise Price and Profit

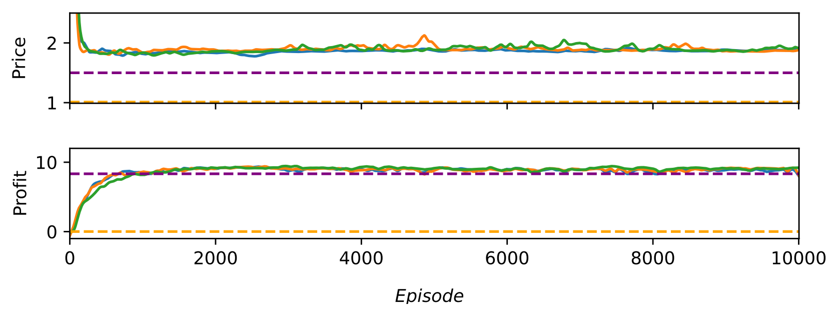
Figure E.4: Mixing the time series and profit-related rewards to train the agents.

E.4 Supervision V5

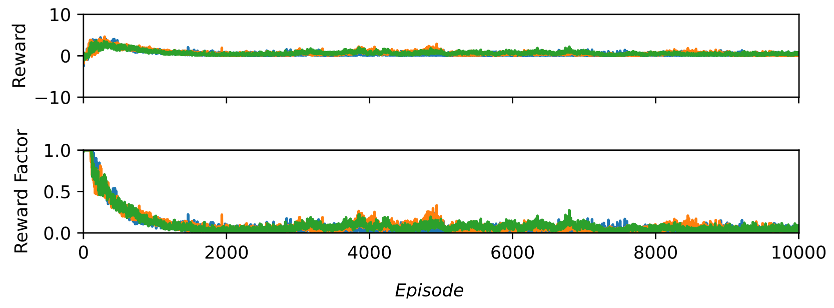
In this experimental setup, we improve upon the previously mentioned conditional distribution of the supervision reward. Specifically, we simplify the calculation by punishing agents only if they act collusively and thereby earn a positive profit:

$$f(r_t^i, r_s(i,t)) = \begin{cases} r_t^i \cdot r_s(i,t) & \text{if } r_t^i \geq 0 \\ r_t^i & \text{if } r_t^i < 0 \end{cases} \tag{E.2}$$

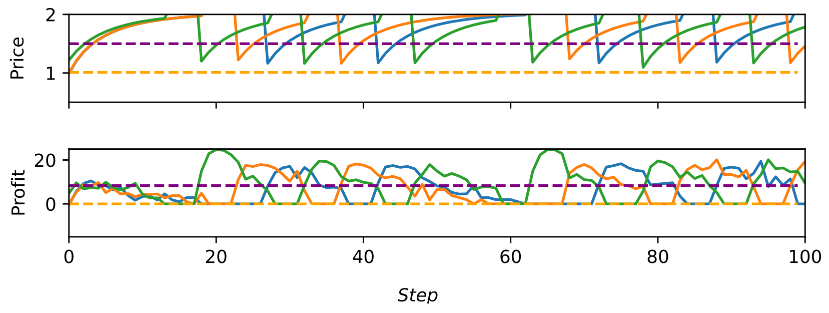
When using classification as the main metric to calculate the supervision reward factor $r_s(i, t)$ (cf. Figure E.5), we did not observe a significant difference compared to runs that only rewarded r_t^i. This suggests that the agents attempted to maximize profit to a value of zero, as achieving a higher profit would introduce randomness and confusion. Furthermore, we tested the same setup with DQNs (cf. Figure E.6). In this case, the agents exhibited even more confusion, resulting in random actions that could potentially yield a decent reward factor. However, as they seldom achieved a positive reward, they did not benefit significantly. Based on these insights, we implemented sparse supervision rewards as described in Chapter E.



(a) Price and Profit

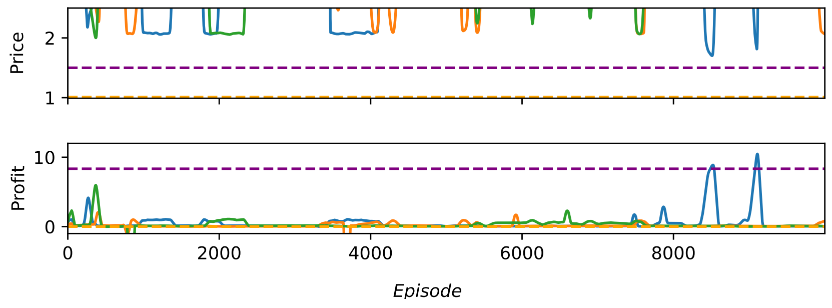


(b) Rewards and Supervision Reward Factor

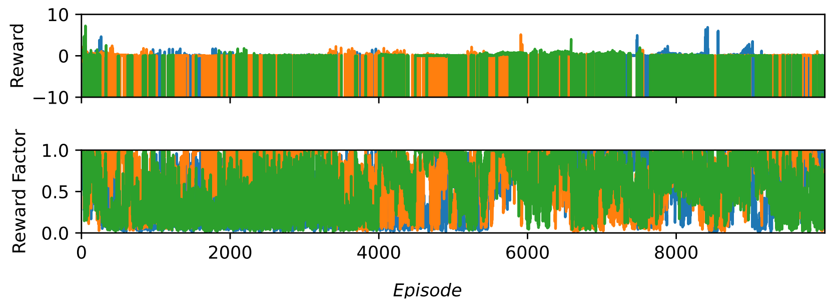


(c) Stepwise Price and Profit

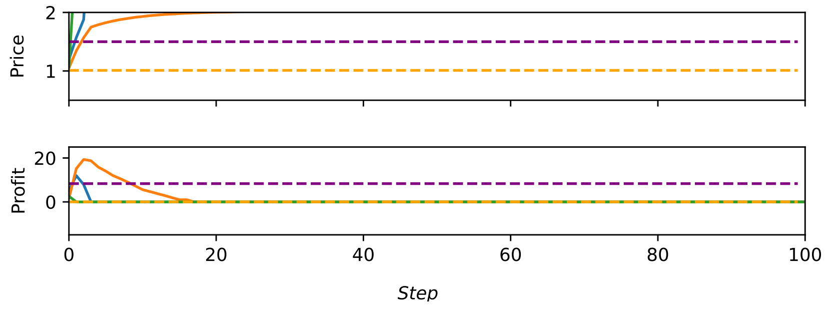
Figure E.5: Using dense supervision reward-based reward, only when positive profit has been achieved.



(a) Price and Profit



(b) Rewards and Supervision Reward Factor



(c) Stepwise Price and Profit

Figure E.6: Using dense supervision reward-based reward, only when positive profit has been achieved with DQNs.