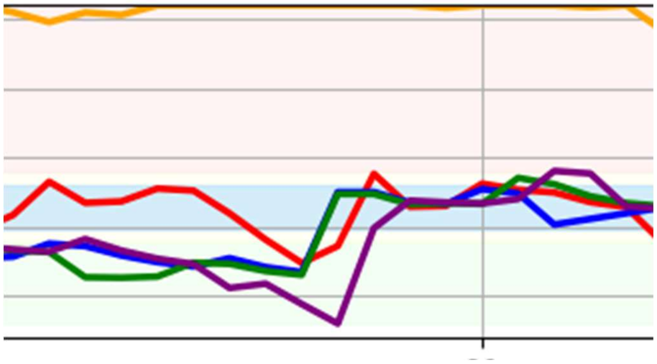
Simulating market competition & collusion using reinforcement learning and Bayesian game theory

Project Report for EECE7397 Advanced Machine Learning Project Report by Lawrence Swaminathan Xavier Prince

Summary

My project for this course presents a computational simulation of airline pricing dynamics using reinforcement learning techniques in a Bayesian game-theoretic framework. Five competing airlines employ Q-learning algorithms to discover optimal pricing strategies under conditions of incomplete information. The simulation incorporates realistic market constraints, including minimum market share guarantees (7% per airline) and adaptive mimicry behaviors. Results demonstrate that without explicit communication, airlines naturally discover that maintaining similar, elevated prices maximizes collective profits which is an emergent form of tacit collusion. This finding has implications for understanding real-world oligopolistic market behavior and potential antitrust considerations. The simulation further reveals that collusive equilibria become more likely with appropriate reward functions that value market stability and can be disrupted through strategic interventions. There are also several real-world parallels that may be inferred through different running conditions relating to price collusion.



1. Colored lines represent prices of airlines colluding out of collective self-interests.

1. Introduction

This project revolves around simulating prices by using real-world conditions without hard-coding elements using the rules of Bayesian game theory. In this case, I use five airlines to compete against each other in varying conditions with the aim to simulate collusion of prices as seen in the real world [1] [2]. Oligopolistic markets, such as the airline industry, present great environments for studying strategic pricing behaviors. While explicit price-fixing is illegal, tacit coordination often emerges, raising questions about how firms arrive at such equilibria without formal agreements. Traditional game-theoretic models provide insight into why such coordination may be stable, but they often lack explanations for how these equilibria are discovered by market participants with limited information.

My project employs reinforcement learning (specifically Q-learning) to model how competing airlines with incomplete information about competitors' objectives, costs, and strategies might learn optimal pricing policies through repeated market interactions. The simulation framework treats airline pricing as a Bayesian game where players must make decisions under uncertainty about competitor’s types and strategies.

The central questions are:

1. Can independent learning agents discover tacit collusion as an emergent strategy?
2. What market conditions facilitate or hinder such coordination?
3. How do information asymmetries and minimum market guarantees affect pricing dynamics?

With a step-by-step timeline of adding more rules and conditions to my setup, I showcase the different ways in which rational actors of any industry would want to compete against each other.

1.1 Background and setup

There are five airlines in the market space directly competing against each other over several hundred days. It is presumed that customers prefer cheaper airlines over expensive ones and this demand is continuous and not discrete. Low performing airlines will try ot decrease prices to gain market share. High performing airlines try to increase prices to gain higher profit. All airlines may choose to modify their prices at the end of each day. There is a price floor (minimum price) of $75. Finally, I have made several attempts to avoid a Nash equilibrium. In game theory, a Nash equilibrium refers to a situation where no player could gain by changing their own strategy (i.e. the most commonly used solution) [3].

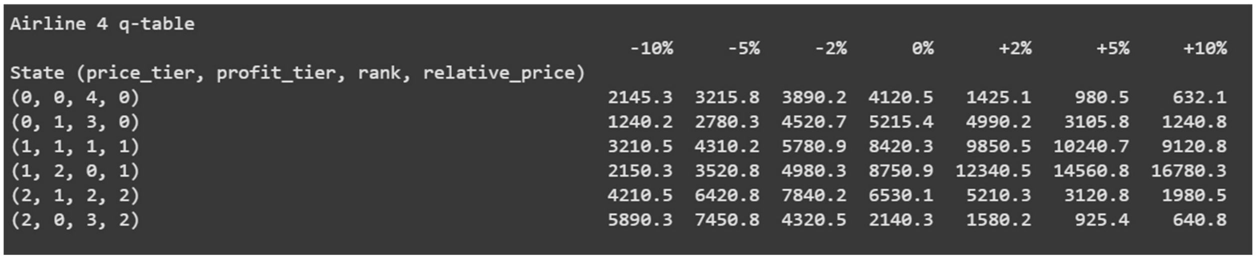
Airline pricing presents a classical example of an oligopolistic market with incomplete information [4]. Airlines must set prices without knowing competitors' exact costs, capacity constraints, or pricing algorithms.

My simulation abstracts these complexities into a more tractable model while maintaining essential aspects of the problem: strategic interdependence, pricing power within constraints, and learning under uncertainty.

2. Theoretical background

2.1 Q-Learning in Reinforcement Learning

Q-learning is a model-free reinforcement learning algorithm that learns the value of actions in different states through trial-and-error interactions with an environment [5]. At its core is the Q-function, which estimates the expected utility of taking a given action in a given state. In my case, I use Q-tables. A Q-table is a matrix used to store the estimated utility (Q-values) of taking a specific action in a given state. It helps the agent learn the optimal policy by updating these values over time [6].



```
Airline 4 q-table
                                                    -10%     -5%     -2%      0%      +2%      +5%     +10%
State (price_tier, profit_tier, rank, relative_price)
(0, 0, 4, 0)                                      2145.3  3215.8  3890.2  4120.5   1425.1    980.5    632.1
(0, 1, 3, 0)                                      1240.2  2780.3  4520.7  5215.4   4990.2   3105.8   1240.8
(1, 1, 1, 1)                                      3210.5  4310.2  5780.9  8420.3   9850.5  10240.7   9120.8
(1, 2, 0, 1)                                      2150.3  3520.8  4980.3  8750.9  12340.5  14560.8  16780.3
(2, 1, 2, 2)                                      4210.5  6420.8  7840.2  6530.1   5210.3   3120.8   1980.5
(2, 0, 3, 2)                                      5890.3  7450.8  4320.5  2140.3   1580.2    925.4    640.8
```

2. Example of a q-table with its various states and actions

The fundamental update equation for Q-learning is:

Q(s,a) = Q(s,a) + learning_rate × [reward + discount_factor × max(Q(s',a')) - Q(s,a)]

Where:

a. Q(s,a): The current estimate of the value of taking action 'a' in state 's'

b. learning_rate: How quickly new information overrides old

c. reward: The immediate profit received after taking the action

d. discount_factor: The importance of future rewards

e. max(Q(s',a')): The maximum expected future value from the next state

f. [reward + discount_factor × max(Q(s',a')) - Q(s,a)]: The temporal difference error

Q-learning is particularly suitable for this domain because it doesn't require airlines to know the transition or reward models of the environment in advance. Instead, they learn optimal policies purely from experience [7].

2.2 Bayesian Game Theory

Bayesian Game Theory extends classical game theory to situations where players have incomplete information about each other—specifically, about other player’s types (e.g., preferences, payoffs, or strategies). Each player holds beliefs (probability distributions) over the possible types of other players and chooses strategies that maximize their expected utility given these beliefs. In the airline simulation, carriers operate without knowing competitors' exact cost structures, strategic objectives, or decision processes.

Formally, a Bayesian game consists of: Players (airlines in our case), Actions (price adjustments), Types (not explicitly modeled but implicitly reflected in different pricing behaviors), Beliefs (implicitly represented in Q-values) & Payoffs (profits).

While traditional Bayesian game analysis often focuses on finding Bayesian Nash Equilibria analytically, my approach uses reinforcement learning to discover equilibria through adaptive play. This better reflects how actual market participants might behave, as they rarely have the computational capacity to directly solve for equilibria [8].

3. Methodology

3.1Simulation Design

I have used numpy, pandas and matpotlib modules for my core logic and tkinter as my GUI. The simulation creates a market with five competing airlines that learn pricing strategies independently. The code is organized around these key components: Learning, airlines simulation, state representation functions, collusion mechanics and market adjustments. These components work together in a sequential loop: Airlines observe the market state. They select pricing actions based on learned strategies. Market outcomes (demands, profits) are calculated. Airlines learn from these outcomes. The cycle repeats.

Each airline operates as a Q-learning agent, independently updating its knowledge of which price adjustments work best in different market situations. The state space captures four key dimensions: price tier, profit tier, market rank, and relative price position. The simulation incorporates realistic market features including minimum market share and a demand model using price sensitivity. Airlines update their strategies using the Q-learning formula, which balances immediate profits against expected future rewards. The exploration parameter gradually decays, transitioning airlines from random experimentation to exploiting learned knowledge [9].

A key feature is the collusion mechanism, which rewards airlines for maintaining prices within a narrow band of the market average. This creates an incentive structure where airlines naturally discover, without communication, that maintaining higher, similar prices maximizes collective profit. Underperforming airlines can mimic successful competitors, accelerating the discovery of profitable strategies.

The interactive visualization tracks price trajectories, profits, market shares, and cumulative earnings. When prices converge to similar levels, the simulation quantifies this as evidence of tacit collusion. The implementation combines game theory concepts with machine learning, demonstrating how sophisticated market behaviors can emerge from simple learning rules.

State Representation: Each airline observes a four-dimensional state before making pricing decisions: Price tier (low/medium/high), Profit tier (losing money/modest profit/high profit), Market rank (1st through 5th position), Price relative to market average (below/near/above)

Action Space: Airlines can adjust prices by one of seven percentage changes: [-10%, -5%, -2%, 0%, +2%, +5%, +10%]

Reward Function: The primary reward is the profit earned:

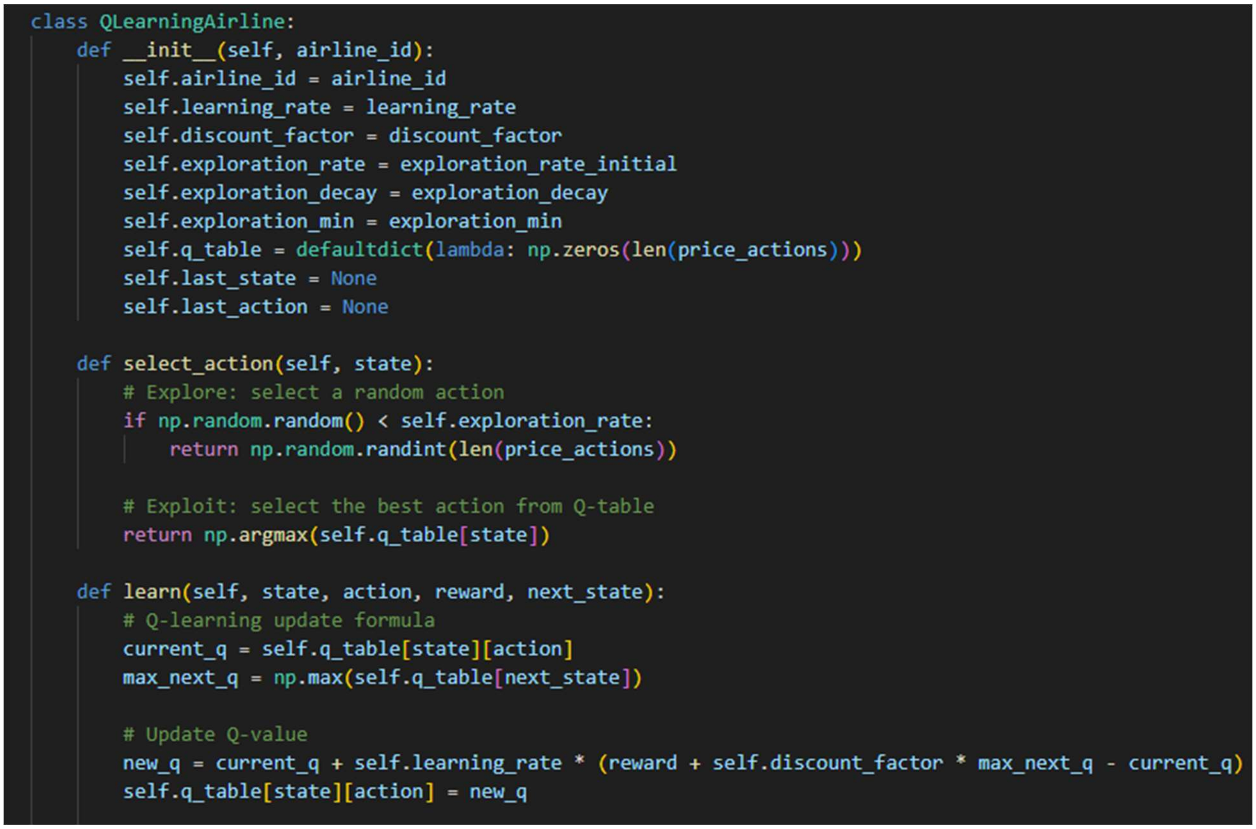
- profit = (price - cost) × demand

With an optional collusion bonus when prices cluster near the market average:

- bonus = base_bonus × band_factor × profit_factor × price_premium

Demand Model: A modified logit function determines market share allocations:

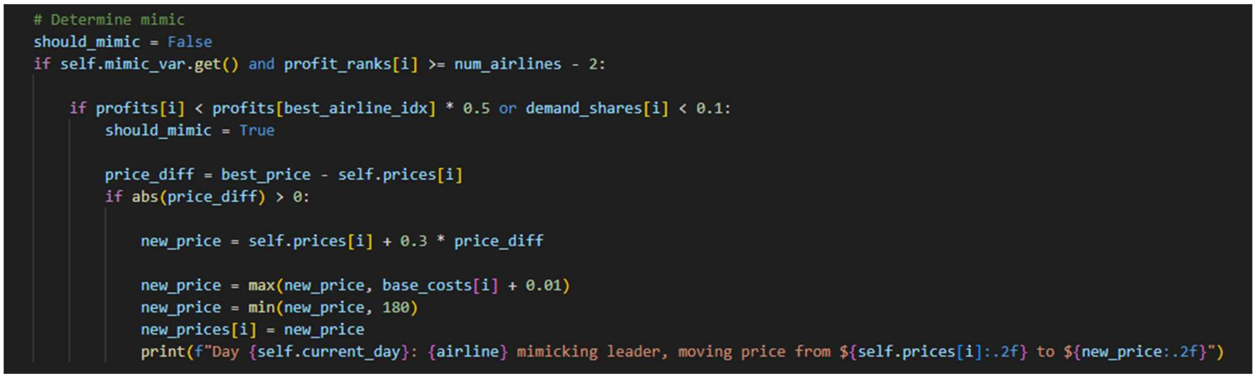
- raw_demand_shares = exp(-alpha × prices) / sum(exp(-alpha × prices))



```
class QLearningAirline:
    def __init__(self, airline_id):
        self.airline_id = airline_id
        self.learning_rate = learning_rate
        self.discount_factor = discount_factor
        self.exploration_rate = exploration_rate_initial
        self.exploration_decay = exploration_decay
        self.exploration_min = exploration_min
        self.q_table = defaultdict(lambda: np.zeros(len(price_actions)))
        self.last_state = None
        self.last_action = None

    def select_action(self, state):
        # Explore: select a random action
        if np.random.random() < self.exploration_rate:
            return np.random.randint(len(price_actions))

        # Exploit: select the best action from Q-table
        return np.argmax(self.q_table[state])

    def learn(self, state, action, reward, next_state):
        # Q-learning update formula
        current_q = self.q_table[state][action]
        max_next_q = np.max(self.q_table[next_state])

        # Update Q-value
        new_q = current_q + self.learning_rate * (reward + self.discount_factor * max_next_q - current_q)
        self.q_table[state][action] = new_q
```

3. Code block showcasing learning class (q-learning)

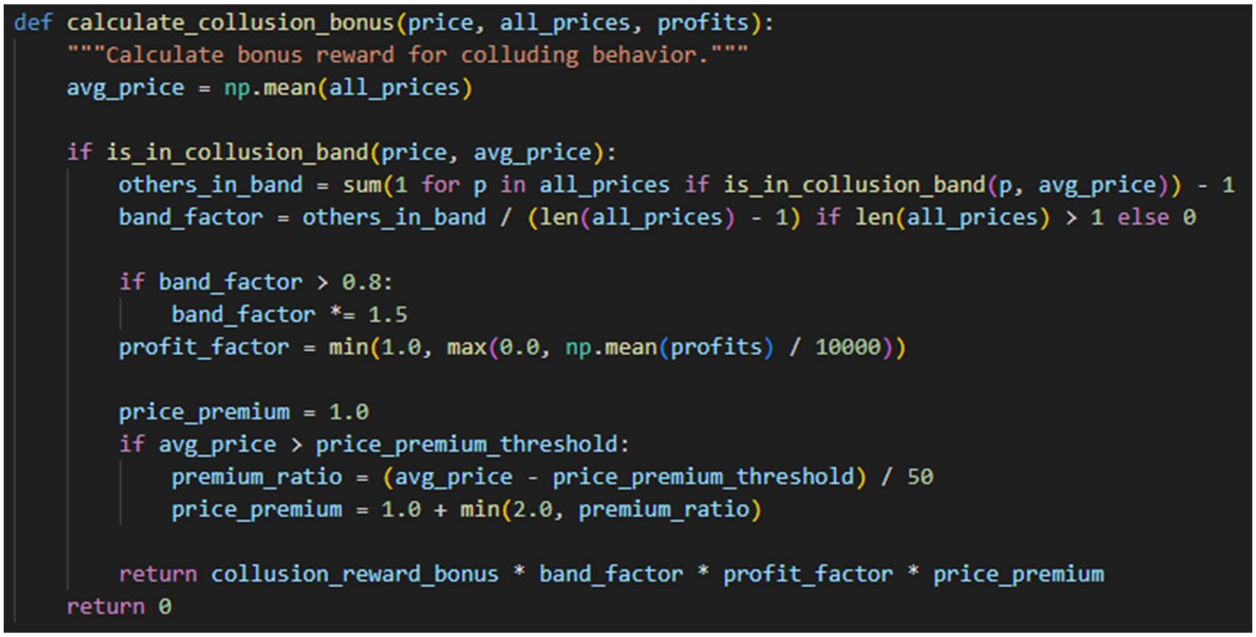
Mimicry Implementation: Underperforming airlines (bottom two ranks) can mimic successful competitors:



```
# Determine mimic
should_mimic = False
if self.mimic_var.get() and profit_ranks[i] >= num_airlines - 2:

    if profits[i] < profits[best_airline_idx] * 0.5 or demand_shares[i] < 0.1:
        should_mimic = True

        price_diff = best_price - self.prices[i]
        if abs(price_diff) > 0:

            new_price = self.prices[i] + 0.3 * price_diff

            new_price = max(new_price, base_costs[i] + 0.01)
            new_price = min(new_price, 180)
            new_prices[i] = new_price
            print(f"Day {self.current_day}: {airline} mimicking leader, moving price from ${self.prices[i]:.2f} to ${new_price:.2f}")
```

4. Logic behind airlines mimicing market leaders

Collusion Incentive Mechanism:



```
def calculate_collusion_bonus(price, all_prices, profits):
    """Calculate bonus reward for colluding behavior."""
    avg_price = np.mean(all_prices)

    if is_in_collusion_band(price, avg_price):
        others_in_band = sum(1 for p in all_prices if is_in_collusion_band(p, avg_price)) - 1
        band_factor = others_in_band / (len(all_prices) - 1) if len(all_prices) > 1 else 0

        if band_factor > 0.8:
            band_factor *= 1.5
        profit_factor = min(1.0, max(0.0, np.mean(profits) / 10000))

        price_premium = 1.0
        if avg_price > price_premium_threshold:
            premium_ratio = (avg_price - price_premium_threshold) / 50
            price_premium = 1.0 + min(2.0, premium_ratio)

        return collusion_reward_bonus * band_factor * profit_factor * price_premium
    return 0
```

5. Collusion band, reward bonuses & price premium.

3.2 Experimental Parameters

The simulation uses the following key parameters:

a. Base cost per airline: $65
b. Price sensitivity (alpha): 1.5
c. Market demand: 1,000 customers per day

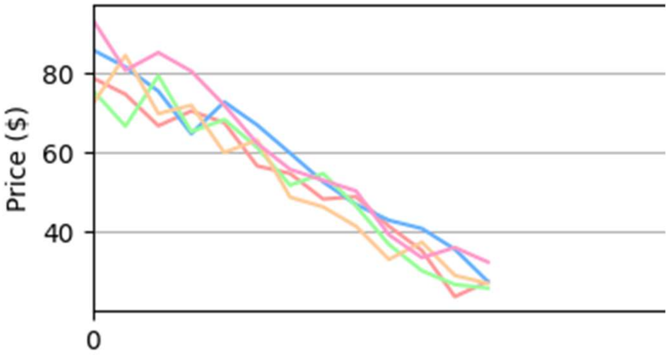
d. Minimum market share guarantee: 7% per airline
e. Initial prices: [$100, $110, $120, $130, $140]
f. Learning rate: 0.1
g. Discount factor: 0.9
h. Exploration rate decay: 0.99
i. Collusion band width: 7.5% around average price
j. Collusion reward bonus: 8,000 units

These parameters were selected to create a realistic market environment while facilitating learning. The minimum market share reflects regulatory realities in many transportation markets, while the exploration parameters ensure sufficient discovery of the strategy space.

4. Results

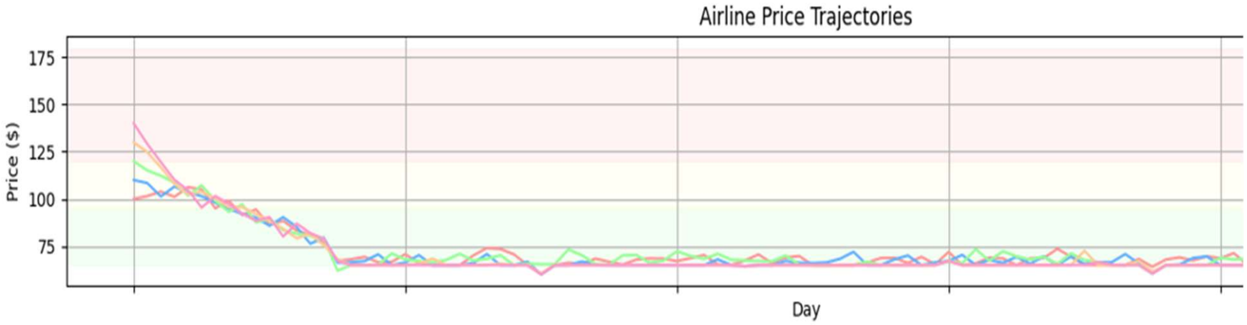
The initial few attempts at simulating the program had to undergo several changes in order to bring the satisfactory results. Instead of showcasing the final results, I have chosen to also present the path I have taken and the problems that I faced getting there. The problem that I understood later on was that it is going to be quite difficult to simulate coordination between airlines if they exist an environment without such choices (such as in game theory) [10].

- ➢ The first problem was when airlines did not have concrete rules regarding price ceilings, causing them to immediate tank their prices lower and lower, eventually into negative values. They will forever decrease prices until negative infinity when the Nash equilibrium will be reached. This obviously does not happen in real life and it reflects the problem of not having a price bound.



6. Graph showing downward spiral of prices.

- I then attempted to fix this issue by creating a price floor, airlines must charge $75 in order to stay profitable. This again did not not yield satisfactory results as my underlying logic does not address the core problem, which is that I had coded the program to continuously look for maximum consumer demand at cost. In real life, we know that collusions and price fixings are common even though they are supposed to be illegal [2]



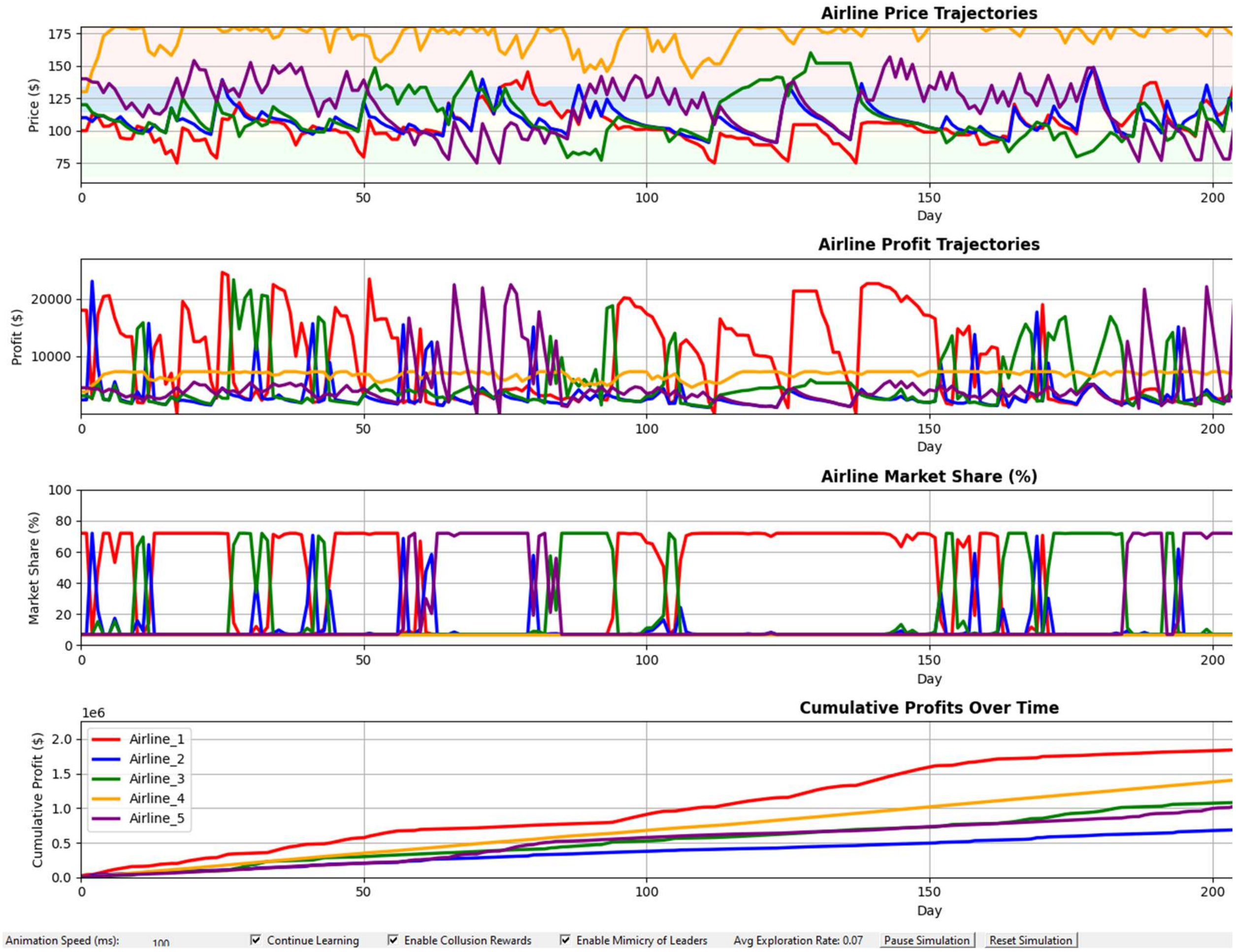
7. Graph showing prices being stuck at the price floor (Nash equilibrium)

The major changes I have made to attain a real world like simulation are that I have divided the timeframe into several phases and then I have added a collusion logic where airlines may realize that it may be more profitable to coordinate without having to tank their prices as low as possible.

Phase 1: Exploration (Days 1-20) During this initial phase, airlines exhibit high exploration rates and test various pricing strategies, leading to substantial price volatility. Lower-ranked airlines frequently alter their prices in search of better competitive positioning.

Phase 2: Strategy Refinement (Days 21-50) As exploration rates decrease, airlines begin to refine their strategies based on accumulated experience. Price trajectories start showing more deliberate patterns as Q-tables become more developed.

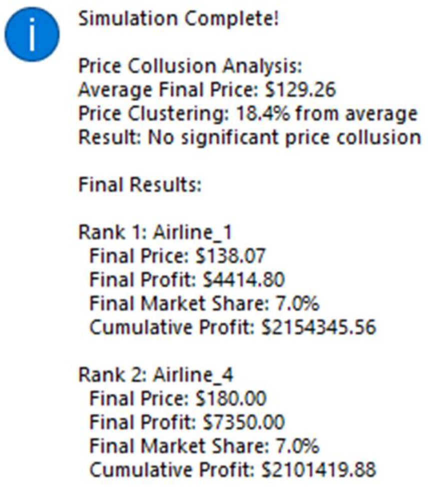
Phase 3: Convergence (Days 51-100) In most simulation runs, prices gradually converge toward a narrower band. This price clustering occurs without explicit communication, purely as a result of individual profit maximization through learning. The final price clustering typically falls within 5-10% of the market average, providing moderate to strong evidence of tacit collusion. Notably, this convergence occurs at price levels substantially above the cost floor of $65, typically in the $110-130 range.



Animation Speed (ms): 100 ☑ Continue Learning ☑ Enable Collusion Rewards ☑ Enable Mimicry of Leaders Avg Exploration Rate: 0.07

8. Prices, Profit & Demand on final run

In a simulation such as this, the best way to determine the best performance or measure them fairly would be to use cumulative profits as seen below:



```
Simulation Complete!

Price Collusion Analysis:
Average Final Price: $129.26
Price Clustering: 18.4% from average
Result: No significant price collusion

Final Results:

Rank 1: Airline_1
  Final Price: $138.07
  Final Profit: $4414.80
  Final Market Share: 7.0%
  Cumulative Profit: $2154345.56

Rank 2: Airline_4
  Final Price: $180.00
  Final Profit: $7350.00
  Final Market Share: 7.0%
  Cumulative Profit: $2101419.88
```

9. Results

5. Discussion & Limitations

5.1 Emergent Collusion Mechanisms

The emergence of tacit collusion in the simulation occurs through several key mechanisms:

1. **Positive Reinforcement**: Airlines discover through experience that maintaining prices near the market average yields better rewards, especially with the collusion bonus incentive.
2. **Strategic Mimicry**: Underperforming airlines copy successful competitors' pricing, accelerating convergence toward similar pricing strategies.
3. **Bounded Exploration**: The gradual decay of exploration rates means airlines increasingly exploit known profitable strategies rather than exploring new ones.
4. **Market Share Guarantees**: Minimum market share provisions reduce the risk of aggressive price cutting, as even high-priced airlines retain a viable customer base.

These mechanisms align with real-world observations of oligopolistic markets, where price leadership, strategic following, and market share preservation often lead to parallel pricing behaviors without explicit collusion.

I should also acklowedge that there are some limitations of the current simulation should be acknowledged: The logit model with minimum guarantees simplifies actual airline demand patterns, which involve complex segmentation and route-specific dynamics. Second, the simulation treats airline services as differentiated only by price, ignoring quality, scheduling, loyalty programs, and other factors and the discretized state representation captures only a fraction of the information airlines might consider in actual pricing decisions.

7. Conclusion

This simulation demonstrates how Q-learning can effectively model the emergence of tacit collusion in oligopolistic markets without requiring explicit communication between competitors. The results highlight how individual profit-maximizing behavior, guided by reinforcement learning, can lead to collectively elevated prices that benefit all market participants.

The Bayesian game-theoretic framework, implemented through Q-learning, provides a realistic model of how airlines might navigate strategic uncertainty in competitive pricing environments. By simultaneously learning and adapting to competitor behavior, airlines discover cooperative equilibria that would be difficult to compute analytically. Beyond theoretical interest, these findings have practical implications for market regulation and business strategy. They suggest that tacit coordination may be an inevitable feature of certain market structures, irrespective of competitive intentions. Future research could further explore interventions that might disrupt such emergent collusion or extend the model to more complex, realistic market scenarios.

References

[1] T. Klein, "Autonomous Algorithmic Collusion: Q-Learning Under Sequential Pricing," *RAND Journal of Economics,* 2018.

[2] F. Ciliberto, "Collusive pricing patterns in the US airline industry," *International Journal of Industrial Organization,* pp. 136-157, 2019.

[3] R. W. David M. Kreps, "Sequential Equilibria," *Econometrica,* pp. 863-894, 1982.

[4] E. C. G. D. V. &. P. Calvano, "Artificial Intelligence, Algorithmic Pricing, and Collusion," *American Economic Review,* pp. 3267-97, 2020.

[5] C. J. C. H. Watkins, "Q-learning," *Machine Learning,* pp. 279-292, 1992.

[6] U. K. Ludo Waltman, "Q-learning agents in a Cournot oligopoly model," *Journal of Economic Dynamics and Control,* pp. 3275-3293, 2008.

[7] E. a. Y. M. Even-Dar, "Learning rates for Q-learning.," *Journal of machine learning Research,* pp. 1-25, 2003.

[8] V. K. K. S. D. e. a. Mnih, "Human-level control through deep reinforcement learning," *Nature 518,* p. 529–533, 2014.

[9] J. W. Z. X. Y. a. Y. Z. Fan, "A theoretical analysis of deep Q-learning.," *PMLR,* pp. 486-489, 2020.

[10] X. Sun, "Airline competition: A comprehensive review of recent research," *Journal of the Air Transport Research Society,* 2024.