

# Automated Pricing Agents in the On-Demand Economy

*Tony Wu  
Anthony D. Joseph, Ed.  
Stuart J. Russell, Ed.*

Electrical Engineering and Computer Sciences  
University of California at Berkeley

Technical Report No. UCB/EECS-2016-57

<http://www.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-57.html>

May 12, 2016



Copyright © 2016, by the author(s).  
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

### Acknowledgement

I would like to thank both Professor Anthony Joseph and Professor Stuart Russell for advising me on this paper. I am grateful for Professor Joseph's support not only on this research project but on my previous research projects in the UC Berkeley SCRUB Lab. I am also thankful for Professor Russell's advice on the reinforcement learning aspects of this paper and teaching a wonderful CS294-125 Human Compatible AI class. I would also like to thank family and friends for their continuous support. I also thank my coworkers at Dray for supporting me through completing my Masters degree alongside startup work. I am excited to integrate Dray's marketplace data with the pricing algorithms explored in this paper.

---

# Automated Pricing Agents in the On-Demand Economy

by Tony Wu

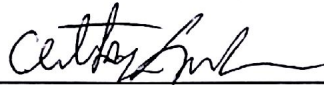
---

## Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences,  
University of California at Berkeley, in partial satisfaction of the requirements for the  
degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

### Committee:



---

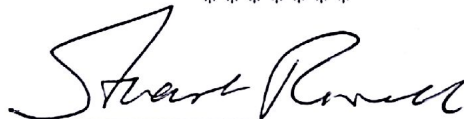
Professor Anthony Joseph  
Research Advisor

May 10, 2016

---

(Date)

\*\*\*\*\*



---

Professor Stuart Russell  
Second Reader

5/11/16

---

(Date)

# Automated Pricing Agents in the On-Demand Economy

Copyright 2016  
by  
Tony Wu

## Acknowledgments

I would like to thank both Professor Anthony Joseph and Professor Stuart Russell for advising me on this paper. I am grateful for Professor Joseph's support not only on this research project but on my previous research projects in the UC Berkeley SCRUB Lab. I am also thankful for Professor Russell's advice on the reinforcement learning aspects of this paper and teaching a wonderful CS294-125 Human Compatible AI class. I would also like to thank family and friends for their continuous support. I also thank my coworkers at Dray for supporting me through completing my Masters degree alongside startup work. I am excited to integrate Dray's marketplace data with the pricing algorithms explored in this paper.

## **Abstract**

Automated Pricing Agents in the On-Demand Economy

by

Tony Wu

Master of Science in Electrical Engineering and Computer Science

University of California, Berkeley

We present this thesis to study and build automated pricing engines that work in on-demand economies. The task of determining market prices is especially challenging in an on-demand marketplace because of frequently fluctuating supply and demand. An automated pricing agent provides reactive real-time prices. To evaluate different types of automated pricing models, we build an on-demand marketplace simulation. By using research on human behavior, we create customer and supplier agent models that interact within the simulation. Afterwards, we run market experiments using various pricing algorithms. We introduce the concept of applying reinforcement learning to generate dynamic prices, and we evaluate our reinforcement learning agent's performance relative to that of other simpler pricing algorithms. Performance is measured not only in terms of profit, but also in metrics that determine customer and supplier retention.

# Contents

<b>Contents</b>	<b>i</b>
<b>1 Market Overview</b>	<b>3</b>
1.1 Marketplaces . . . . .	3
1.2 Market players . . . . .	4
1.3 Differences from other marketplaces . . . . .	4
1.4 Rise of automated pricing systems . . . . .	5
1.5 Relevant literature review . . . . .	6
<b>2 Pricing Simulation Setup</b>	<b>8</b>
2.1 Overview . . . . .	8
2.2 Agent preferences . . . . .	9
2.3 Population models . . . . .	10
2.4 Action models . . . . .	11
2.5 Modeling different agent types . . . . .	12
2.6 Modeling cost . . . . .	13
2.7 Setting the initial price . . . . .	14
<b>3 Dynamic Pricing Algorithms</b>	<b>16</b>
3.1 Overview . . . . .	16
3.2 Static pricing . . . . .	16
3.3 Proportional pricing . . . . .	16
3.4 Batch update pricing . . . . .	17
3.5 Reinforcement learning . . . . .	17
<b>4 Short Term Simulations</b>	<b>19</b>
4.1 Overview . . . . .	19
4.2 Surge pricing . . . . .	19
4.3 Comparison . . . . .	21
<b>5 Long Term Simulations</b>	<b>30</b>
5.1 Overview . . . . .	30

5.2 Comparison . . . . .	31
<b>6 Future Work</b>	<b>38</b>
<b>7 Conclusion</b>	<b>40</b>
<b>Bibliography</b>	<b>41</b>



# Chapter 1

## Market Overview

### 1.1 Marketplaces

Many types of marketplaces exist in the world today, each with their own pricing strategies. This paper focuses on pricing in the “on-demand” economy. We can define the on-demand economy as a marketplace where companies provide fulfillment of customer demand via the immediate provisioning of goods and services. On-demand services have become increasingly popular with improvements in Internet availability, mobile technologies, and real-time operations. It is common for supply and demand to vary greatly in these marketplaces. Since the goods and services are delivered immediately, companies often have to price their commodities on the spot depending on market factors at the time. Therefore, prices are required to be adaptive and respond to real-time changes in supply and demand.

Many traditional pricing strategies exist today that would not perform well in the on-demand economy. In the traditional retail space, commodity supply and prices are preset and generally stationary throughout the day. This model is especially prevalent in stores with inventory, such as grocery stores and fashion retail. Prices are adjusted infrequently, commonly on a day-to-day basis. When customer demand changes, there is little the seller can do to adjust accordingly in a timely fashion. In fact, the increasing rate of markdowns in the retail industry has been caused by growing demand uncertainty[15].

Contract pricing is another widely used pricing model. Under contract pricing, the price of a commodity is normally determined by certain factors written in a contract. An example would be taxi prices. By taking a taxi ride, you are agreeing to pay a certain amount per mile or time with certain additional costs such as fuel. Prices determined by contracts only change if specified in the contract. Therefore, there is no way for prices to react to unforeseeable changes in the marketplace. Contracts are also usually legally binding and difficult to rewrite and renegotiate.

Bid pricing strategies are also commonplace across multiple industries. The first-price sealed-bid auction is how we traditionally think about bidding, where the highest bidder wins and pays what they bid. The second price auction is a more stable auction system where

the highest bidder pays the second highest bid. This system is famously used by Google for its search advertisement[27][13]. However, a bidding system would incur too much overhead in an on-demand marketplace. The bidding process causes a delay in pricing adjustments and therefore a delay in responses to marketplace changes.

## 1.2 Market players

Many industries contain on-demand marketplaces. The airline ticketing industry is a more traditional example. Airlines sell a supply of seats using similar dynamic pricing techniques. Although the fulfillment of goods is not as immediate, airline ticket sales must deal with the same fluctuating demand that on-demand companies face[2]. In addition to the airline industry, the trucking industry has a very on-demand component in its spot brokering segment. Spot brokering is the immediate pricing of a freight lane for a customer. A lane is defined as a commonly traveled route. An example would be the I-5 freight lane going from Seattle to Los Angeles. This bypasses the traditional bidding system in favor of a more expedited experience. However, it is often hard to generate spot prices, as lane prices usually depend on a variety of factors including supply, demand, type of good shipped, carrier preferences, shipper preferences, etc[35].

Now, newer industries exist that have popularized on-demand services while incorporating more data-centric techniques into their pricing, such as machine learning. There are transportation companies such as Lyft and Uber, food delivery companies such as Caviar and Munchery, and general service providers such as TaskRabbit. Although all these companies offer different goods and services, they all face the same problem of fluctuating supply and demand and the need to respond to these changes in real time[32].

## 1.3 Differences from other marketplaces

The difference between these on-demand marketplaces and other more traditional marketplaces is the need for real time feedback and adjustments. Due to the fact that these companies tout immediate fulfillment, they do things at a much quicker pace than other slower moving industries. As a result, their markets are much more sensitive to changing supply and demand. Other pricing models are too slow to adjust to these constant changes. Companies that operate in the on-demand space must remain agile[9]. As a result, their pricing systems must also be agile enough to react to changing environments.

The cost of not setting the right price is also much costlier in an on-demand marketplace. On-demand commodities are ephemeral. If a commodity is not being fully utilized, there are many associated sunk costs, such as fuel, and maintenance costs. The same sunk costs apply across airline, on-demand transportation, and trucking industries. If an airplane flies with empty seats, those seats are a sunk cost for the airline. If a truck driver or Uber driver drives with an empty vehicle, the driver is losing money on fuel and car maintenance costs. Since

on-demand transactions are so frequent, if prices are slow to adjust, costs could magnify within the course of hours.

Traditionally, prices are seen as reactions to changing market environments. On the other hand, on-demand companies have effectively employed pricing as a means to directly influence supply and demand. For example, Uber raises prices during peak hours to increase driver supply and defer customer demand[17]. Therefore, an automated pricing agent should be able to take advantage of its ability to influence behaviors of market agents.

## 1.4 Rise of automated pricing systems

Automated pricing systems have been shown to be the answer to the variance that on-demand companies have faced. Our definition of an automated pricing system is an agent that is able to respond to data provided by the environment (current supply, demand, etc.) and adjust the price accordingly. Overhead is reduced, because it should be able to observe data and take action without human supervision. As urban data becomes more available through initiatives such as Open Data, more companies and institutions have been incorporating these logistics into their operations[5][20].

Both Uber and Lyft use a proprietary form of an automated pricing system. Prices are generated on the spot, and fares for the same route can vary greatly from minute to minute. These prices usually take supply and demand into account. One example is Uber's surge pricing[17]. During periods of high demand, it is common for users to see prices go up.

Automated pricing systems have been used successfully in other industries. Since its introduction as a low cost airline in 1992, RyanAir has revolutionized the airline industry by being profitable even with relatively low prices[3]. Another one of the first low price airlines, People Express, also offered low prices but could not compete. People's Express offered flat rate low cost flights and experienced tremendous growth in the 1980s. However, it eventually lost business as a result of more flexible pricing strategies employed by competitors. As a pioneer of dynamic pricing strategies, RyanAir has remained competitive by taking advantage of "latent demand". Prices for RyanAir tickets change constantly, and almost always get higher as the flight date approaches. This is based on the expectation that customers who buy last minute tickets are more willing to bear the costs. We can also see how this dynamic pricing has also influenced consumer behaviour, where passengers seek to find days where tickets might be lowest, sometimes booking many weeks or months in advance. Currently, almost all airlines employ some variation of this pricing model. Since the average price of dynamic pricing is higher than a flat low-cost rate, Ryan-Air and other airlines can remain competitive and profitable while still offering budget prices.

Other traditional industries have begun adopting a more dynamic approach to pricing as well. Disney just announced that it plans to introduce demand-based pricing at its theme parks[12]. Event ticket sellers face the same sunk costs as on-demand industries, and they are now also changing their prices based on estimated supply and demand. Notable examples include concert ticket sellers such as Stubhub and major sports leagues[10][25].

## 1.5 Relevant literature review

There is an abundance of prior research that uses dynamic pricing to tackle price optimization in the face of uncertain supply and demand. The papers that do not have a large marketplace dataset rely on simulated data. Most of these papers have emulated marketplace interactions using a multi-period finite horizon problem[16][19][33][36]. In this paper, we use this multi-period finite horizon model to build a simulation that operates in a similar manner. The scope of prior research ranges from industry specific solutions to a broad application of dynamic pricing. This paper falls in the middle, because we examine a simulation specific to on-demand transportation, but we preserve the re-usability of our simulation by avoiding industry specific marketplace parameters. Most general papers remain flexible by only incorporating overarching marketplace factors, such as supply, demand, and price.

Dynamic pricing is relevant in a multitude of industries. Kim (2014) has explored the use of dynamic pricing to manage electricity grids. Shih (2003) shows that dynamic pricing can be used to change user behavior in voice and data usage by deferring spikes in demand[31]. Bertsimas (2006) shows that dynamic pricing can be generalized to any problem that involves congestion pricing[7]. A lot of prior work in dynamic pricing has been done as a response to yield management and perishable goods[16]. These industries face challenges that are also applicable to the on-demand marketplace.

Prior simulation papers propose varying techniques to emulate customer and supplier behavior and the way behavior reacts to changing prices[30][11][18]. Raju (2006) proposes a customer segmentation of “captives” and “shoppers”, where “captives” are mature, loyal customers and “shoppers” are more price sensitive[30]. Dimicco (2003) models unpredictability in buyer and seller behavior and the effects that pricing has over supply and demand in conjunction with advertising and brand perception[11]. Haws (2006) examines dynamic pricing and its effect on consumer fairness perceptions and temporal sensitivity[18]. In this paper, we introduce agent traits that generalize buyer and seller behavior. We do not delve into advertising and brand perception, but we are able to model similar interactions in price sensitivity, temporal sensitivity, and agent sentiment towards price. Numerous economic papers delve into customer satisfaction and retention, but they do not employ the same dynamic pricing algorithms we use in this paper[8].

Many papers have translated the dynamic pricing problem to a Markov decision process problem[23][6]. More specifically, the dynamic pricing problem is considered a partially observed Markov decision process, as agents do not necessarily have full observability of the marketplace[6]. Miller (1968) proposes a finite horizon Markov decision process where only a finite number of actions (prices) are allowed. We employ a similar technique in this paper, where we define a possible range of prices and discretize that interval into actions. Another valid way of formulating the problem is to represent it as a game between market participants[23].

Reinforcement learning has been a solution of interest for determining dynamic prices. Since the complete model of a marketplace is not known, model-free techniques such as Q-learning are effective in determining optimal pricing actions. Most other papers, such as

Pednault (2004), use profit as the main metric of success, but do not consider other factors such as customer or supplier happiness[29]. They also tend to overlook customer and supplier retention, which may be negatively or positively influenced by price. These are all factors we measure when analyzing the final action-value models learned by our reinforcement learning agent.

Competition in the marketplace has been modeled in different ways. Hu and Wellman (1998) show a multi-agent RL situation where two agents repeatedly play a game where both agents have full observability and converge to optimal strategies[22]. In reality, marketplace agents operate with partial observability[24]. Kephart (2000) examines the competition between two RL agents with partial observability. Kephart (2000) also models competition between an RL agent and game-theory agents with simpler search strategies. Introducing competition between automated agents can lead to unexpected price wars and effects on consumers and suppliers. We explain how we might translate our results to a multi-competitive landscape in the future work section.

## Chapter 2

# Pricing Simulation Setup

### 2.1 Overview

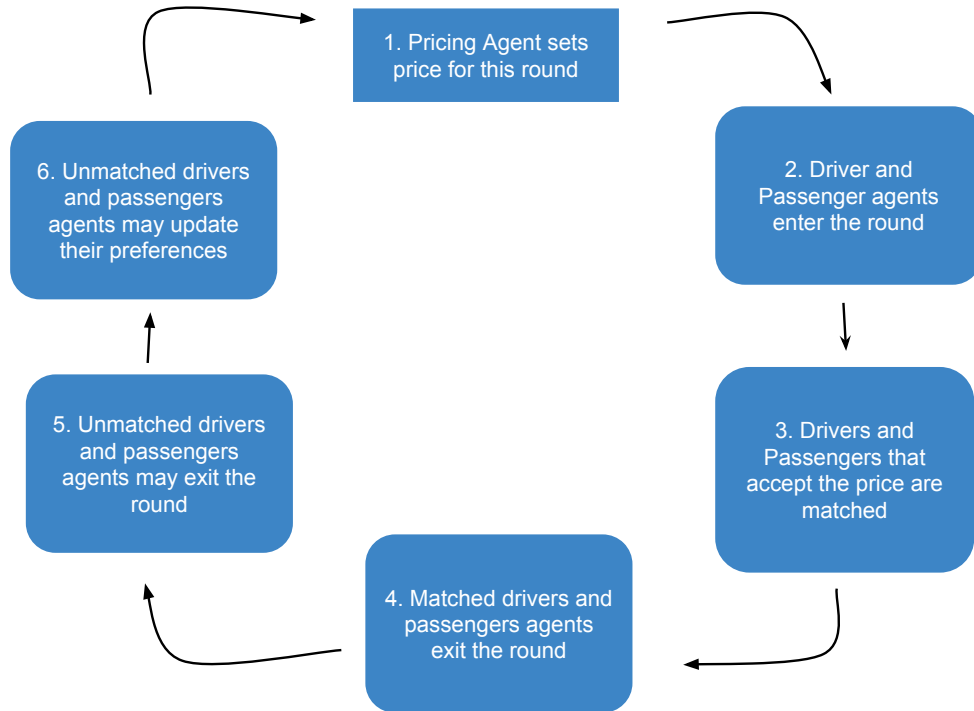
In order to evaluate the performance of different pricing models, we built a simulation that models the dynamic environment an on-demand market. We chose to model a situation similar to an on-demand ride requesting service such as Uber or Lyft. The agents in this simulation are the passengers, the drivers, and the automated pricing agent. The purpose of the automated pricing agent is to set a price for a lane, which is a ride from point A to point B at time  $t$ . Both driver and passenger agents may choose to accept this price, and each driver-passenger pair constitutes a match. Each match will result in some amount of revenue for our pricing agent proportional to the market price at the time.

For each agent, we must build a preference model. This preference model determines each agent's actions and their utilities. For both drivers and passengers, we must also generate a population of agents with a diverse set of preferences that reflect a real world population.

The simulation must also model the passage of time. Therefore, the simulation will consist of rounds, where each round represents the passage of some unit of time. In each round, we must update a set of variables that describe environment. These include the current price, agents that match, agents that exit, and each agents preference models. A general overview of a round in our simulation can be seen in Figure 2.1. Specifics on how and why we update market factors are provided later in this section.

The point of the simulation is to model how price variations should react to and affect supply and demand in a certain lane. An example of a lane is a route from Downtown Oakland to UC Berkeley. This simulation does not account for multiple different lanes or geolocation of agents. We talk about how we might tweak this simulation to model many different geographical lanes in the future work section.

Figure 2.1: Visualization of a matchmaking round



## 2.2 Agent preferences

Agent preferences are variables that affect an agent’s behaviour. We specifically chose preferences that would be relevant to real human behavior. The variables are as follows:

### Passenger and driver variables

- Preferred price,  $p_{preferred}$ : The price at which the agent would accept an offer for this lane. Everyone has a price that they would pay/take to use the service. For drivers, this is a price where if they were offered any lower price, they probably would not take the offer. For passengers, this is a price where if they were offered a higher price, they likewise probably would not take the offer.
- Price adjustment tolerance,  $p_{tolerance}$ : Determines how much the agent’s preferred price changes for each round they are active and still unmatched. We imagine agent price preferences to be dynamic. This trait models a sense of urgency or fluidity in agent preferences. For example, if passengers do not get matched immediately, they might

be willing to pay more as time goes by as their urgency increases. On the other side, drivers might care less and less about how much they are getting paid as they get more desperate to make money.

- Round tolerance,  $R_{tolerance}$ : How many rounds the driver stays active without being matched. This trait is here to model how frustrated agents may switch off the service and use another alternative if they do not find an immediate match or offer that is satisfactory. An example of this is if you think Uber prices are too high and after 5 minutes, you leave the Uber app to use Lyft. These agents that exit do not exit permanently and may return for subsequent rounds.
- Satisfaction,  $S$ : How likely the driver is to stay or leave the system permanently. This trait represents the lifetime satisfaction of an agent using the service. The lower an agent's satisfaction, the more likely they are to leave the service permanently. Satisfaction is increased as the agent gets matched, and is decreased as an agent does not get matched.
- Income preference,  $I$ : Threshold on how much a driver agent needs to make over a period of time. (e.g. \$100 over 7 rounds). Drivers tend to drive to meet a minimum level of income. The main cause of Uber and Lyft driver churn has come from drivers realizing that they cannot make enough money on the platform. In our simulation, after a certain period of time, driver's compare the money they've made to their income preference. If their preference is not met, they leave the simulation permanently.

## Price agent variables

- Market price,  $p_{market}$ : The current price offered for this lane.

## 2.3 Population models

To create a population of agents, we need to create a number of agents with a wide variety of these preferences. For this simulation, we decided to use a multivariate Gaussian based population model. While real data is not exactly multivariate Gaussian, the Gaussian distribution is a good approximation because of the central limit theorem. We establish an average preferred price, price adjustment tolerance, round tolerance, and satisfaction for the population. Then we generate agents so that each preference follows a Gaussian distribution in the population. This gives us a range of agents with a diverse set of preferences. We can set the averages of each variable to some reasonable value that we derive from market research.



## 2.4 Action models

With these preferences set, we can create models on how agents are likely to act based on their preferences. There are a few actions that active agents can take each round:

### 1. Accept an offer

We can model the probability that an agent will accept an offer given the price by using a probit distribution. The probit distribution is the integral of a standard normal distribution. Probit distributions are useful for determining decision probabilities that rely on soft thresholds. The threshold in this case is the agent's preferred price. We can justify using a soft threshold by proposing that even though the underlying decision of accepting an offer may have a hard threshold, the precise location of that threshold is subject to random Gaussian noise. If the integral of a normal distribution is defined as:

$$\Phi(x) = \int_{-\infty}^x N(0, 1)(x)dx$$

then the probability of a passenger accepting an offer given a price is:

$$P(\text{accept} \mid p_{\text{offer}}) = \Phi((-p_{\text{offer}} + p_{\text{preferred}})/\sigma)$$

and the probability of a driver accepting an offer given a price is:

$$P(\text{accept} \mid p_{\text{offer}}) = \Phi((p_{\text{offer}} - p_{\text{preferred}})/\sigma)$$

where  $\mu$  is the agent's preferred price and  $\sigma$  determines the width of the threshold region.

We can see here that the higher the price, the less likely a passenger is to accept the offer, and more likely a driver is to accept the offer. If an agent does not get matched to an offer, actions 2 to 4 may occur.

### 2. Adjust preferred price

For each round that an agent does not get matched, they may be willing to adjust their price tolerance based on their price tolerance. For passengers, this means they are willing to pay a higher price, and for drivers, they are willing to drive for a lower price. For simplification purposes, we can just add or subtract a percentage of the preferred price to an agent's preferred price each round. Here is our update function for passengers, for drivers we just reverse the sign of the operation:

$$p_{\text{preferred}} = p_{\text{preferred}} * (1 + p_{\text{tolerance}})$$

### 3. Become inactive

If an agent becomes unmatched for a number of rounds exceeding their round tolerance, they will become inactive. This just means that an agent has left the match-making process temporarily.

$$\text{state} = \begin{cases} \text{inactive} & R_{\text{unmatched}} > R_{\text{tolerance}} \\ \text{active} & \text{otherwise} \end{cases}$$

## 4. Leave the system

If an agent's satisfaction becomes too low, they may permanently leave the system. Similar to accepting an offer, we can set the probability that an agent will leave given their satisfaction using the probit distribution:

$$P(\text{leave} | S) = \Phi((S_{\text{average}} - S)/\sigma)$$

where  $S$  = satisfaction and  $S$  ranges from 0 to 1.0.  $\sigma$  determines the width of the threshold region. We set  $S_{\text{average}}$  to be 0.5 in our simulations. Each round, an agent's satisfaction can go down or up depending on whether or not they get matched.

$$S = \begin{cases} \min(S + S_{\text{gain}}, 1) & \text{agent matched} \\ \max(S - S_{\text{loss}}, 0) & \text{agent unmatched} \end{cases}$$

In addition, drivers may leave the system if their income is less than their income preference. Simulation specifics about income preference are provided in the long-term simulations section.

## 2.5 Modeling different agent types

There are many combinations of these preferences. We can tweak these preferences so that we model different types of agent populations in the marketplace. Some notable examples are as follows:

## 1. Average passenger and driver

An agent with average values across all traits.

## 2. Urgent passenger

An urgent passenger would be an agent that has a high price tolerance but a low round tolerance. This is someone who has to get somewhere quick with little regard for price.

## 3. Critical passenger

A passenger with low price tolerance, low round tolerance, and high satisfaction loss. This is someone who is skeptical of what the service provides, similar to a first time user, and they are easily dissatisfied and prone to leaving the platform.

## 4. Critical driver

This is similar to a critical passenger. This is a driver with a low price tolerance, low round tolerance, and high satisfaction loss.

## 2.6 Modeling cost

Our goal in this paper is to maximize profit while maximizing metrics that define driver and passenger satisfaction. Profit is defined as the difference between revenue and cost. Therefore, we must have a way to model costs in our simulation. There are two costs that we are focused on. Customer acquisition costs and driver acquisition costs. Ideally, our agent should achieve higher profits if it is incentivized to decrease costs. A simple model of representing cost is to generate a constant for acquisition costs and loss costs. For example, for each customer we acquire we add one unit of customer acquisition cost to our cost pool. To come up with quantitative values for these costs, we can look at costs that currently exist in the industry.

### Customer acquisition cost

As the time of writing this paper, Uber and Lyft currently offer \$20 dollars off your first ride[34]. We can use this as an approximation of our customer acquisition cost. Of course, there are other factors such as marketing costs and human labor that increase the customer acquisition cost, but these are too hard to quantify on a general basis.

### Driver acquisition cost

The cost of acquiring a driver tends to be higher than the customer acquisition cost. Acquisition costs for drivers include marketing, driver training, and other driver services such as insurance. We can get a rough approximate of driver acquisition costs by looking at the current driver referral bonus. Lyft is willing to pay drivers in San Francisco \$750 if they refer a new driver[26]. Therefore, in our simulations, we decide to set the driver loss cost to be similar in magnitude.

### Modeling competition

We model competition by introducing a customer and driver loss cost. This cost represents the advantage a competitor gains when a customer or driver permanently leaves our platform. These costs also incentivize increasing driver and passenger satisfaction and retention. We make customer and driver costs equal to their acquisition costs. As a result, driver loss costs are relatively higher than customer loss costs, and we emphasize retaining drivers over customers when given the choice. This is a common problem currently seen in the ride-sharing and trucking industries. On-demand companies have seen that customer demand has grown, but disgruntled drivers have been leaving as a result of low pay, poor treatment, and other factors. This has resulted in a driver shortage for both Uber, Lyft, and the trucking industry as a whole. Since drivers are difficult to replace and retrain, it makes sense that we should put a higher value on driver retention. We describe more complex ways to model competition in the future work section.

## Modeling enter rate using price sensitivity

An agent's usage of on-demand platforms is also sensitive to price. Drivers, especially, are very sensitive to the current market price. Platforms, such as Uber, have tried to manipulate the supply of drivers by surging prices. Drivers react by flocking to areas with higher prices. We model the probability that a driver will enter given a market price by using a probit distribution, which increases the chances of new drivers entering the system when there is a higher price.

$$P(\text{enter} | p) = \Phi((-p + p_{\text{preferred}})/\sigma)$$

where  $p$  = current market price and  $p_{\text{preferred}}$  is the driver's preferred price.

## 2.7 Setting the initial price

We must decide the initial market price for our pricing engine. There are multiple ways we can do this:

### Random Guess

Alternative one is to use a random number and have our pricing engine eventually converge to the right value.

### Competitor Pricing

Another alternative is to get an approximation of what people are willing to pay for a certain lane. We can get a good sense of how much people are willing to pay to use our service by looking at the competitive alternatives. An obvious alternative would be a traditional player in this field. In the on-demand transportation case, this traditional player would be the taxi industry. The taxi pricing model is quite straightforward. They charge a certain dollar amount per unit of distance or unit of time, whichever is greater. An example is the NY taxi service. We have included a general overview of their pricing model below[28]:

- The initial charge is \$2.50.
- Plus 50 cents per 1/5 mile or 50 cents per 60 seconds in slow traffic or when the vehicle is stopped.
- There is a daily 50-cent surcharge from 8pm to 6am.
- There is a \$1 surcharge from 4pm to 8pm on weekdays, excluding holidays.
- Passengers must pay all bridge and tunnel tolls.
- Please tip your driver for safety and good service.

- Other NY specific surcharges

As a result, we can apply the NY taxi model to our lanes to determine the initial price. This approach is good for approximation, but inaccurate since we may be offering a very different service. Taking an Uber is a different experience from taking a taxi, so it might make sense to offer a different price point.

### **User Surveys**

We can also conduct surveys of passengers and drivers to determine a rough estimate of prices they would like to pay. However, due to human nature, passengers results might tend to be skewed toward lower prices while drivers would be biased toward higher prices.

### **Historical Data**

If we have historical data on how much people are willing to pay to use a service, we could just use the averages of that data. In our case, we did not have Uber or Lyft data specifically. However, we did use the New York taxi data collected from 2010 to 2013 by UC Berkeley Professor Alex Bayen [1]. This data showed that the average NY taxi fare was \$13.95. The initial market price that we ended up setting for simulation is \$15.00. This keeps our relative values for cost and revenue proportional to real life values.

# Chapter 3

## Dynamic Pricing Algorithms

### 3.1 Overview

After setting our initial base price, it is up to our pricing algorithm to change the market price in reaction to dynamic supply and demand trends. The goal of the pricing algorithm is to modify the price to ensure profit maximization. Here we discuss various dynamic pricing algorithms that we will later use in our simulations.

### 3.2 Static pricing

This is the most simple pricing algorithm, where the price never changes.

$$p = p_{base}$$

### 3.3 Proportional pricing

In proportional pricing, our price is equal to the base price multiplied by a constant factor times the ratio between supply and demand.

$$p = p_{base} * k * \frac{d}{s}$$

Proportional pricing is derived from the economic principle that when there is high demand, people are willing to pay more for a good and service. In addition, when demand is high and pay is good, more suppliers are willing to enter the market. The hope is that proportional pricing solves the imbalance of supply and demand in dynamic marketplaces.

### 3.4 Batch update pricing

Batch update pricing offers multiple prices centered around our base price. Our approach is to offer a Gaussian distribution of prices each round. As offers get accepted, we shift our base price towards the prices that result in a match. This is a relatively greedy approach that uses recent matches to update our round price.

$$p_{base} = p_{base} - \alpha * (p_{base} - p_{matched})$$

where  $\alpha$  is the learning rate

As time increases, we can decrease the standard deviation of our offerings, which allows for less experimentation but more exploitation. We can also decrease our learning rate so that our base price converges faster.

### 3.5 Reinforcement learning

An autonomous agent should be able to observe an environment and learn how to act in the environment given a success metric. This method is known as reinforcement learning. In our relevant literature review, we mention other studies that use reinforcement learning to provide dynamic pricing[29][7][4]. Similarly, we can model our simulation as a Markov decision process (MDP). Then, we plan on training our agent using iterations of Q-learning. After training, we can see how this RL agent reacts in our market simulations.

#### Markov Decision Processes

Markov Decision Processes consists of:

1. A set of possible world states, S
2. A set of possible actions, A
3. A reward function, R(s,a)
4. A transition function which describes each action's effect in each state, T(s,a,s')

To apply this to our simulation, we need to discretize our simulation into states and actions. For our pricing agent, an action is equivalent to setting the market price as some price  $p$ . We can limit our actions by specifying a reasonable minimum and maximum price for the action space. For our purposes, the minimum price can be set as 0. Uber caps their prices at 9.9 times the base price, so we can use that as our maximum price. We bucket our price levels into intervals of 2 to obtain a reasonably sized action and state space.

In our state space, each state can be described using three state variables:

1. Market price, M

2. Number of drivers, D
3. Number of passengers, P

We assume in our simulations that the number of drivers and passengers never exceeds 500. In addition, we discretize our market price state into intervals of 2. Our reward,  $R(s,a)$ , is defined as the profit gained by choosing a certain price for the given state.

### Q-Learning

Since we do not know the transition functions, we cannot create a complete model of the MDP. Therefore, we can use a model-free approach such as Q-learning. Q-learning allows us to learn an action-value function that gives us the expected utility of taking a given action in a given state. Q-learning is also able to factor in future rewards into this action-value function. The general formula for Q-learning is as follows:

$$\begin{aligned}
 &\text{initialize } s_0 \\
 &\text{for } t = 1, 2, 3 \dots \\
 &\quad \text{choose an action } a_t, \text{ let's say } \epsilon\text{-greedy w.r.t. } Q \\
 &\quad \text{execute } a_t \\
 &\quad Q(s_t, a_t) = (1 - \alpha_t)Q(s_t, a_t) + \alpha_t [R(s_t, a_t, s_{t+1}) + \gamma \max_a Q(s_{t+1}, a)]
 \end{aligned} \tag{3.1}$$

Since we want our agent to maximize profits, the reward function will equate to the profit gained by taking action  $a$  from state  $s$ .

Computation time to solve an MDP grows exponentially with the number of state variables. In this case, the state space is greatly simplified. Therefore, we can consider the computational intensity of solving the MDP exactly. Based on previous definitions, our simulation's MDP has a state space of size  $\{M\} * \{D\} * \{P\}$ , which comes out to be  $5000 * 5000 * (price_{max}/2)$ . Each state has  $price_{max}/2$  possible actions. Given enough time, this MDP can be exactly solved. Because of time restrictions, the Q-learning agents in this paper use action-value models that have not completely converged to optimal values. With additional training, the Q-learning results in this paper can achieve better performance.

### Hyperparameters

We set the learning rate  $\alpha$  to be  $\alpha_t = \frac{K}{t+K}$  where  $K$  is some large value relative to  $t$ , and  $t$  represents the learning iteration. Although we have not shown it, we know this algorithm converges as long as we visit all the states with probability  $P(\cdot) > 0$ [14].

For the discount factor  $\gamma$ , values closer to 0 makes our agent "short sighted" and value short term rewards. Values closer to 1 will cause the agent to strive for long term reward. We chose to set the discount factor to 0.99.



# Chapter 4

## Short Term Simulations

### 4.1 Overview

Depending on the length of our simulation, we can create short or long term simulations. Each round of our simulation represents an hour, and short term simulations last 20 rounds. Short terms simulations are meant to model the dynamic fluctuations in supply and demand that happen over the course of a day. Short term simulations are simpler to model, because we do not have to model long term customer acquisition costs and customer loss. We can be solely focused on revenue maximization and reacting to more drastic trends in supply and demand.

For this simulation, we set the base market price as \$15. The preferred price for both drivers and passengers is \$15. Both drivers and passengers have a round tolerance of 2. Drivers and passengers also have a price tolerance of 33.33%, which leads to a preferred price change of \$5 each round.

### 4.2 Surge pricing

Surge pricing is good example of an interesting short term simulation. Surge pricing is currently employed by companies such as Uber and Lyft in response to a sharp increase in demand in a short period of time[17]. Demand usually spikes during certain times of day. Notable examples include the morning rush to work and after concerts and events. To deal with this, pricing agents normally need to surge the price in order to defer the increase of demand and to also increase the supply of drivers. From an economic perspective, this is explained by the Law of Supply and Demand as seen in Figure 4.1.

In our experiment, we have a total of 20 time steps, which represents 20 sequential hours in a day. During each round, we inject 50 additional passengers and drivers. We simulate a demand surge of 150 passengers in the 2nd and 3rd rounds and see how our pricing agents deal with the scenario. We also see how supply and demand react to the changing market

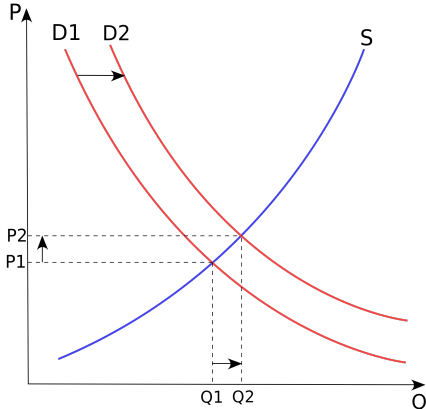


Figure 4.1: Explanation of surge pricing from an economics understanding. D1 and D2 represent demand curves. S represents the supply curve. The P axis represents price. The Q axis represents quantity. P1 and P2 represent price levels. Q1 and Q2 represent quantity levels. The Law of Supply and Demand states that as demand rises from D1 to D2, the price of a commodity increases from P1 to P2. The quantity of the supply must also rise from Q1 to Q2 to match the increased demand.

price. We keep track of important metrics such as matches made per round, revenue per round, total revenue, etc.

**Static Pricing Results**

We first used our static pricing model in our surge simulation. The results are shown in Figure 4.2.

**Proportional Pricing Results**

We experimented with two different approaches to proportional pricing: Uncapped vs. capped proportional pricing.

With uncapped proportional pricing, prices are directly proportional to the ratio of demand and supply. The problem with this approach is that whenever there is a sharp increase in demand, a sharp increase in price follows. However, this price can easily reach a price that no reasonable passenger would be willing to pay. We can already see this frustration in real life, as passengers complain whenever Uber or Lyft enact any type of surge pricing. For example, Uber capped its surge pricing at 9.9x during the immensely busy New Year’s Eve night in 2015. Rides that normally cost around \$20 exceeded \$100. This greatly hurt Uber’s public image and business. To remedy this, in some areas, such as Boston, Uber has put an artificial cap at 2.9x[21].

We emulate this cap in two ways. First, we experiment with setting a lower constant of proportionality in our equation. By setting k as a lower value, we can limit the dynamic

range of our price. The second method is to just set a cap on the pricing agent’s price. In our case, we set the cap to be 2x the base price.

Results for uncapped proportional pricing with  $k=2$  are shown in Figure 4.3. Results for uncapped proportional pricing with  $k=1$  are shown in Figure 4.4. Results for capped proportional pricing are shown in figure 4.5.

### Batch Update Pricing Results

For batch update pricing, we set the learning rate  $\alpha$  to be 0.1. The results are shown in Figure 4.6.

### Reinforcement Learning Results

For reinforcement learning, we went through 50,000 iterations of training. In each iteration of training, the agent went through 20 rounds of the simulation. The results for reinforcement learning pricing are shown in Figure 4.7.

## 4.3 Comparison

Results Overview			
Pricing model	$\hat{R}_t$	$\hat{E}_d$	$\hat{P}_m$
Static	0.65	0.76	0.90
Proportional with $k=1$	0.75	0.82	0.95
Proportional with $k=2$	0.94	1.00	0.99
Proportional with cap at 2x	0.95	0.99	1.00
Batch Update	0.97	0.98	0.99
Reinforcement learning	1.00	1.00	0.95

Table 4.1: An overall comparison of pricing model performance for the short term simulation.  $\hat{R}_t$  = normalized total revenue.  $\hat{E}_d$  = normalized earnings per driver.  $\hat{P}_m$  = normalized proportion of passengers matched

Table 4.1 and Figure 4.8 show a comparison of all the pricing methods. In terms of total revenue, reinforcement learning outperforms all other pricing methods. We can observe a few important factors that seem to correlate to a pricing agent’s success in the surge pricing simulation.

A phenomenon that we see is that there are many exiting passengers after the surge period regardless of pricing model. These are all the passengers who were not matched due to the lack of drivers. It is obvious that there needs to be an increase in price when demand greatly exceeds supply. With price sensitive drivers, increasing the price helps supply match demand. In simulations where the increase in drivers eventually matches the surge in passengers, there

are a large amount of exiting drivers post-surge. This behavior is actually seen in real life. It is common for drivers to drive towards a surge area, but when they arrive, most of the passengers have already left and the surge has ended, which causes dissatisfaction amongst drivers.

The static pricing model is the only model that does not increase price in the surge simulation. As a result, it does not take advantage of the increase in demand, because the number of matches is steady over time.

The simulations also show that drastic price increases are not a solution. In the uncapped proportional model, prices get as high as \$100. As a result, no matches are made in that round, because no passengers are willing to pay that high of a price. This also results in the largest spike of exiting passengers.

The last three pricing models do not suffer from this problem. They respond to the increase in demand, by raising prices to increase supply. As a result, the number of matches increases during the surge and revenue also spikes during the surge.

An interesting thing to note is that even though we did not put a price cap on the batch update and reinforcement learning pricing models, they both never exceed 3x the base price. In fact, the highest surge we see is the 2.5x in the RL model. This is similar to Uber's surge cap of 2.9x in some areas of the United States[21].

Both batch update and RL pricing models reach similar revenue peaks after the surge period. Total revenue for the RL model is higher because it tends to perform better on average at other times during the simulation. With more training, the RL model should be expected to do even better as more state spaces are explored.

In terms of user and customer satisfaction, we can see that the RL model is among the highest in earnings per driver. However, the RL model is also the lowest in terms of proportion of passengers that get matched. The sharp increases in prices that the RL model makes could be increasing profits at the expense of leaving some passengers unmatched.

From our short term simulation results, we can come to a few conclusions. If given the choice, a company should implement the RL pricing model if they are only concerned about profit maximization. The RL pricing model also comes with the benefit that on average, drivers are paid more. This results in long-term driver satisfaction and incentivizes drivers to join the platform. However, we can see that the RL model results in the lowest percentage of passengers that are eventually matched. This could be a problem if customers are sensitive to inconsistent service. If RL prices lead to too many unhappy customers, this can lead to low customer retention and poor public perception. Therefore, a company must determine whether their cost-benefit analysis justifies sacrificing customer satisfaction for higher profits and driver earnings. If not, another pricing model such as the batch update model might be more optimal.

Figure 4.2: Simulated surge results for our static pricing model

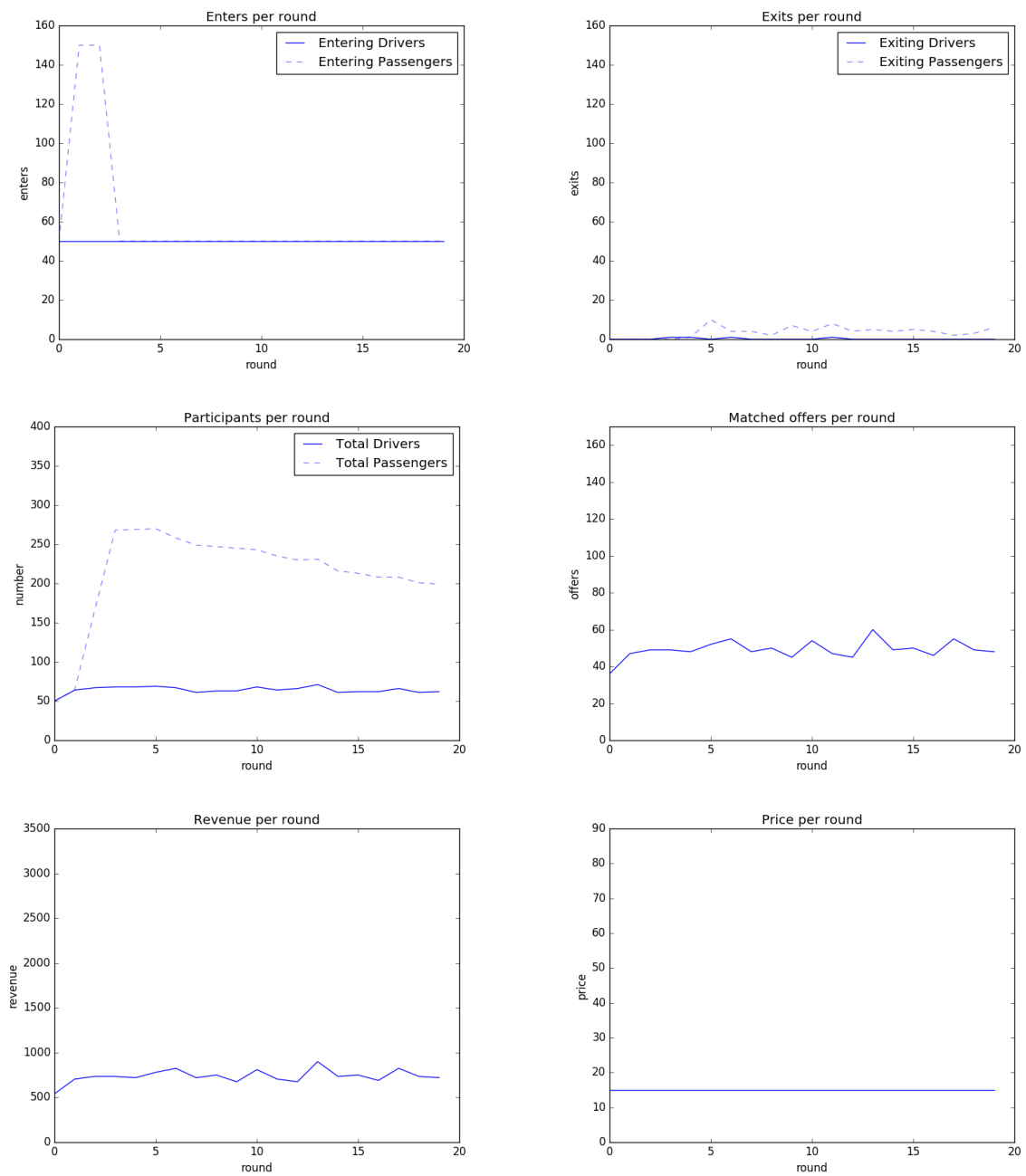


Figure 4.3: Simulated surge results for our proportional pricing model with  $k = 2$

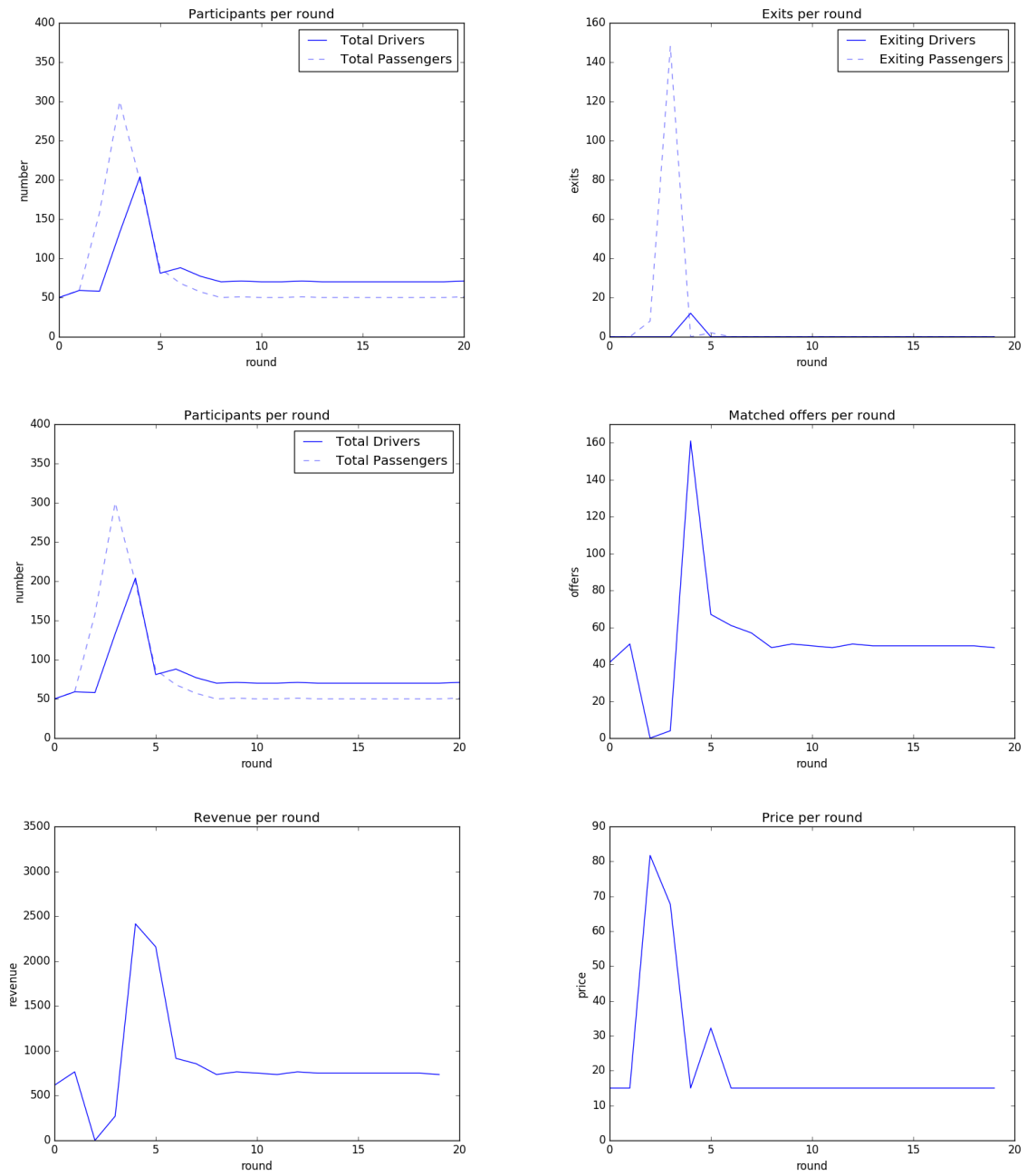


Figure 4.4: Simulated surge results for our proportional pricing model with  $k = 1$

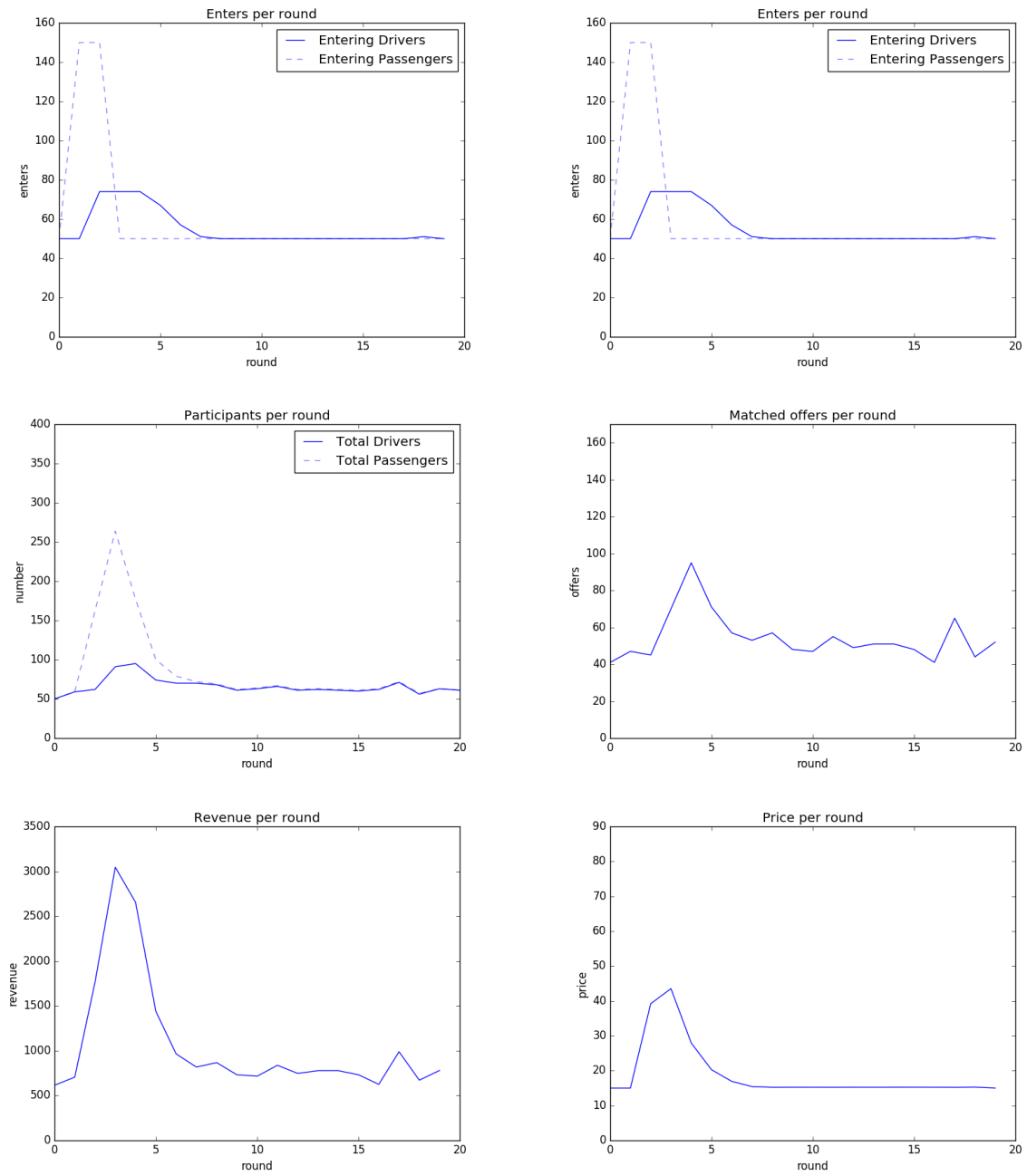


Figure 4.5: Simulated surge results for our proportional pricing model with cap at 2x base price

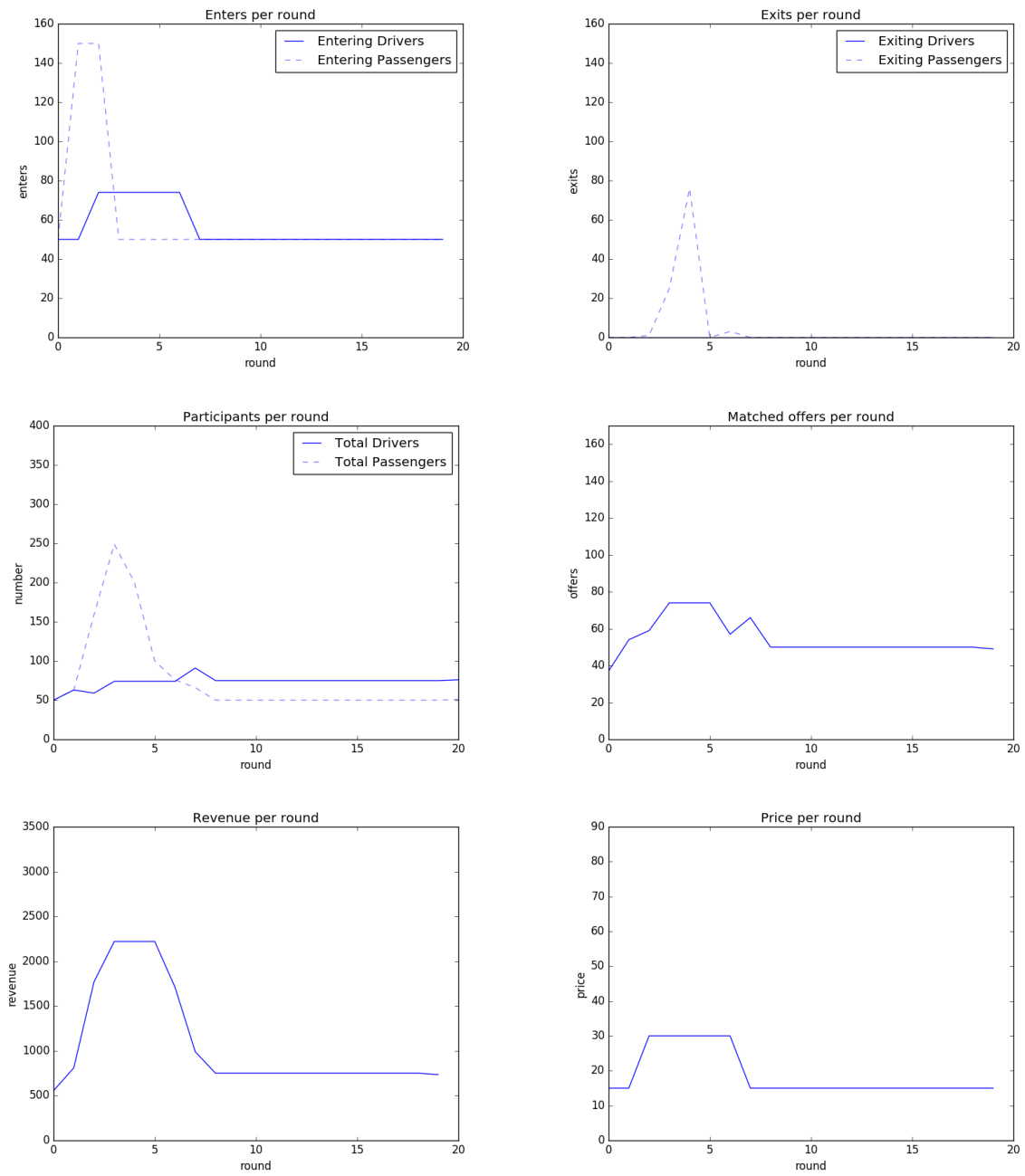




Figure 4.6: Simulated surge reactions to our batch update pricing model

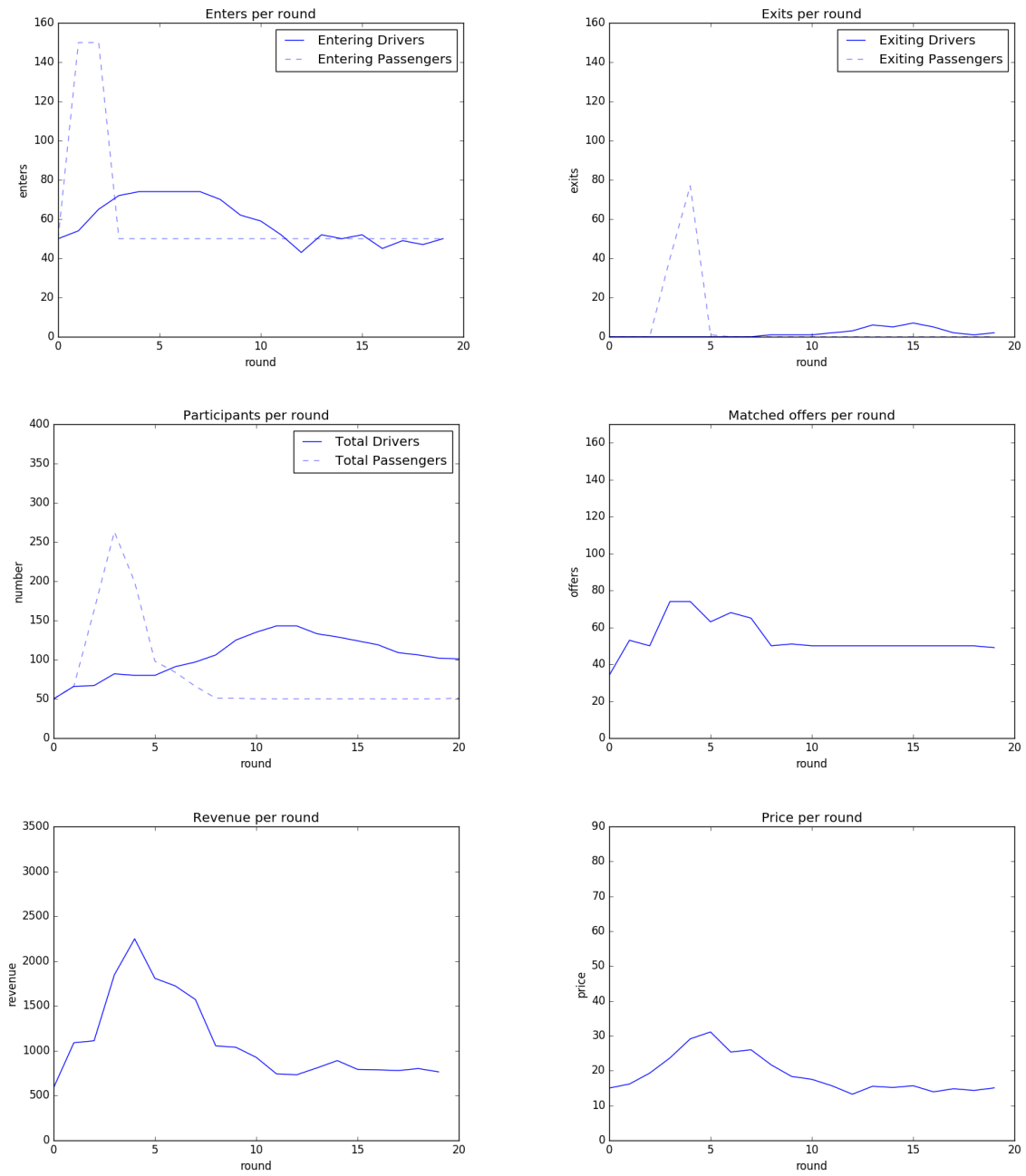


Figure 4.7: Simulated surge results for our reinforcement learning pricing model

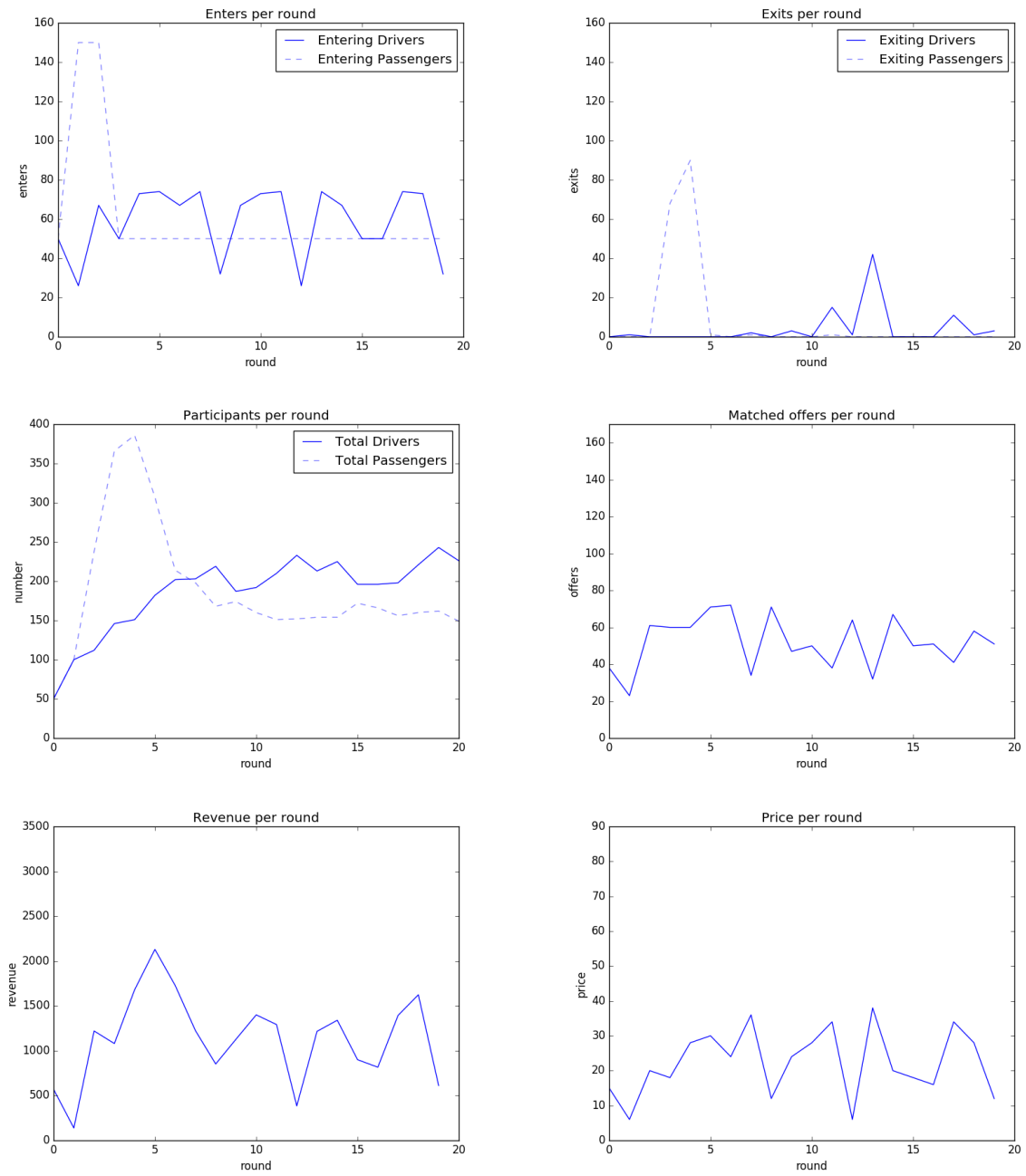
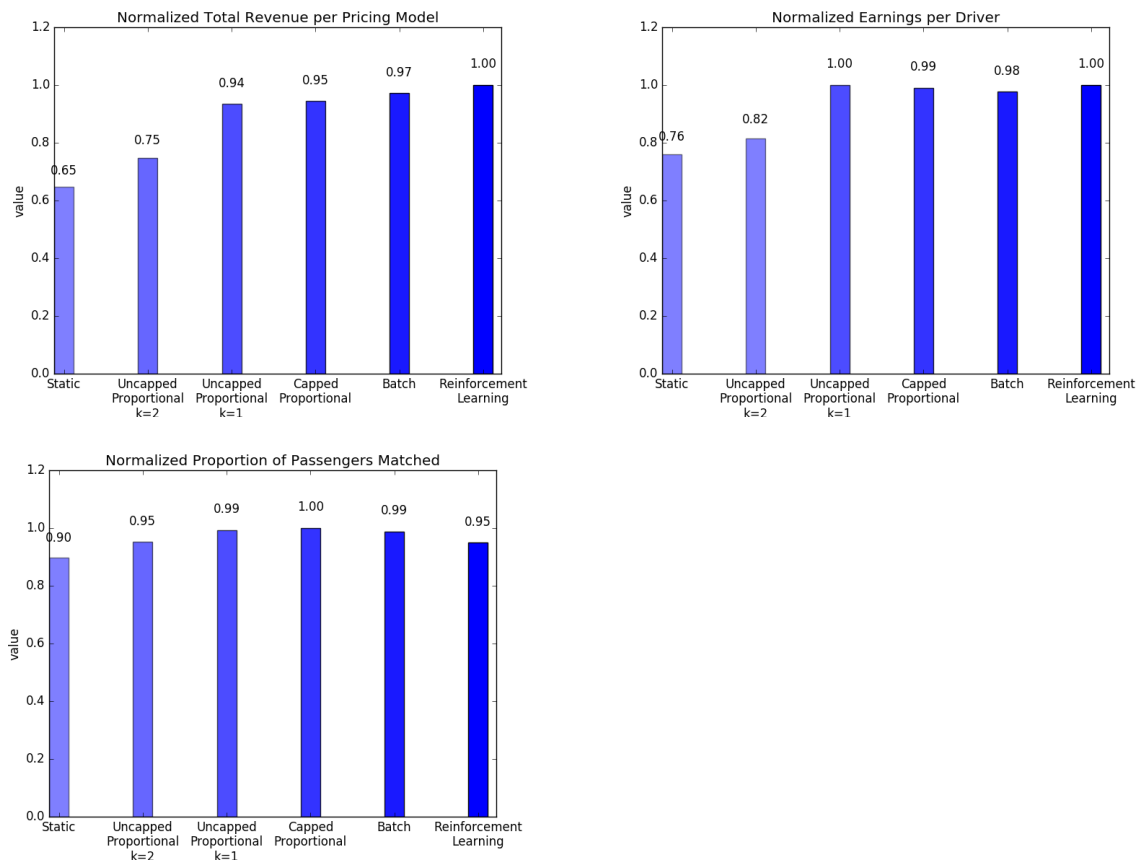


Figure 4.8: Comparison of pricing models in our short term surge simulation



# Chapter 5

## Long Term Simulations

### 5.1 Overview

For our long term simulations, each time step represents an hour. Our experiments last 720 time steps, which is intended to simulate a month's duration. Long term simulations are harder to simulate, since there additional factors that we are interested in keeping track of. We must now factor in acquisition and loss costs for both drivers and passengers. We did not consider this in our short term simulations, because during the course of the day, both costs are negligible.

For customer acquisition, we initiate our simulation with an initial pool of 50 drivers and passengers. We add a number drivers and passengers to our population each round according to a normal distribution with mean 50 and standard deviation of 5. Each customer acquisition increases our cost by \$20, and each driver acquisition increases our cost by \$100. Our population is divided up into active and inactive agents. Each round, agents that are active become inactive if they have not been matched for  $n$  rounds, where  $n$  is their round tolerance. Agents that are inactive may become active with probability 0.5. Agents may also permanently leave the population. The probability that they permanently leave is related to their satisfaction according to the probit distribution as mentioned in Chapter 2. Each time an active agent gets matched, their satisfaction increases by 0.1, and each time an active agent does not get matched, their satisfaction decreases by 0.1. Therefore, if an agent is repeatedly unmatched, they are more likely to defect permanently.

Our drivers also have a weekly income preference of \$900, which equates to an hourly wage of \$22.5. This means that each driver evaluates the amount of money they have made over the past 140 rounds and sees whether or not their income has reached \$45. If not, they leave our system permanently. Depending on the market price, this means that drivers will have be matched at least 3 to 4 times over the course of 7 rounds. For this simulation, we set the base market price as \$15. The preferred price for both drivers and passengers is \$15. Both drivers and passengers have a round tolerance of 2.

### Static Pricing Results

We first used our static pricing model in our surge simulation. The results are shown in Figure 5.1. We can see that the price remains static throughout the whole simulation. Therefore, the pricing agent cannot use price to adjust supply to meet fluctuating demand. We can see a wide gap between active drivers and passengers. This cyclical gap is caused by unhappy passengers leaving because of the lack of drivers, which causes a driver surplus. This absence of passengers causes drivers to leave, which leads to a passenger surplus and a repeat of the whole process.

### Proportional Pricing Results

We decided to set  $k=1$  for our proportional pricing model. In order to calculate price, we only considered the ratio between active passengers and drivers. The results are shown in Figure 5.2. We can see that proportional pricing does a great job of matching supply and demand.

### Batch Update Pricing Results

For batch update pricing, we set the learning rate  $\alpha$  to be 0.1. The results for batch update pricing are shown in Figure 5.3. Batch update pricing tends to maintain higher prices, which causes our simulation to have a high quantity of drivers at all times. There is a constant driver surplus at all times during the simulation, which leads to high passenger match rates, but also higher driver dissatisfaction.

### Reinforcement Learning Results

For reinforcement learning, we went through 50,000 iterations of training. In each iteration of training, the agent went through 720 rounds of the simulation. Results are shown in Figure 5.4. The results from reinforcement learning tend to be quite noisy. This is caused by the large fluctuations of pricing learned by the RL agent. RL pricing does a decent job at matching driver and passenger quantities. Explanations for the large fluctuation in pricing could be that the pricing agent tends to take advantages of situations where it can price higher to maximize profits. Then, it may make drastic decreases or increases in price to reduce the gap in supply and demand that might have resulted from its profit maximizations.

## 5.2 Comparison

Table 5.1 and Figure 5.5 show a comparison of all the pricing methods. The RL model ends up with the highest profit and revenue out of all the models followed by the batch update model, proportional model, and static model.

The proportional pricing model and the static pricing model tend to have higher costs. This cost is mostly due to the fact that both pricing models use prices to increase supply to

Results Overview					
Pricing model	$\hat{P}_t$	$\hat{R}_t$	$\hat{C}_t$	$\hat{E}_d$	$\hat{P}_m$
Static	0.82	0.81	0.85	0.83	1.00
Proportional	0.83	0.82	0.86	0.84	1.00
Batch Update	0.95	0.90	0.99	0.85	1.00
Reinforcement learning	1.00	1.00	1.00	1.00	0.96

Table 5.1: An overall comparison of pricing model performance for the long term simulation.  $\hat{P}_t$  = normalized total profit.  $\hat{R}_t$  = normalized total revenue.  $\hat{C}_t$  = normalized total cost.  $\hat{E}_d$  = normalized earnings per driver.  $\hat{P}_m$  = normalized proportion of passengers matched

match demand. This increases the driver acquisition cost, which is relatively expensive in our simulation.

Table 5.1 and Figure 5.5 show the revenue, total cost, and profit breakdown of each pricing model. The total cost is calculated as follows:

$$C_t = N_{pe} * C_{pe} + N_a * C_a$$

Or in plain text terms: the total cost is equivalent to the number of permanently exiting agents \* the cost of permanently losing an agent + the number of acquired agents \* the cost of agent acquisition.

We can see that the RL model has the highest earnings per driver by far. All other models only reach about 80% of the earnings per driver that the RL model achieves. The RL model has a slightly lower proportion of passengers that get matched. This might be due to the drastic changes in prices that the RL model makes. Similar to the short-term experiment, sharp increases in price might increase profits, but could also cause passengers to leave the system.

From our long term simulation results, we can come to a few conclusions. We can see that the RL model results in the highest profits in both the short-term and long-term simulations. This makes it a flexible pricing model that handles both drastic short-term fluctuations and long-term driver and passenger retention. Therefore, a company focused on profit maximization would be inclined to use a RL pricing model. Again, we see that drivers under the RL pricing model are paid more. The RL model's advantage in driver income is even more noticeable in the long-term simulations compared to the short-term simulations. We also see the same drop-off in proportion of passenger matched. This plus the high costs associated with the RL pricing model are reasons companies might not want to use RL pricing. Since our costs are estimated, real companies might have different cost models, which could influence actual profits. If relative acquisition and loss costs are higher than we estimated, the RL model might not actually be the best in profit maximization.

Figure 5.1: Simulated long term results for our static pricing model

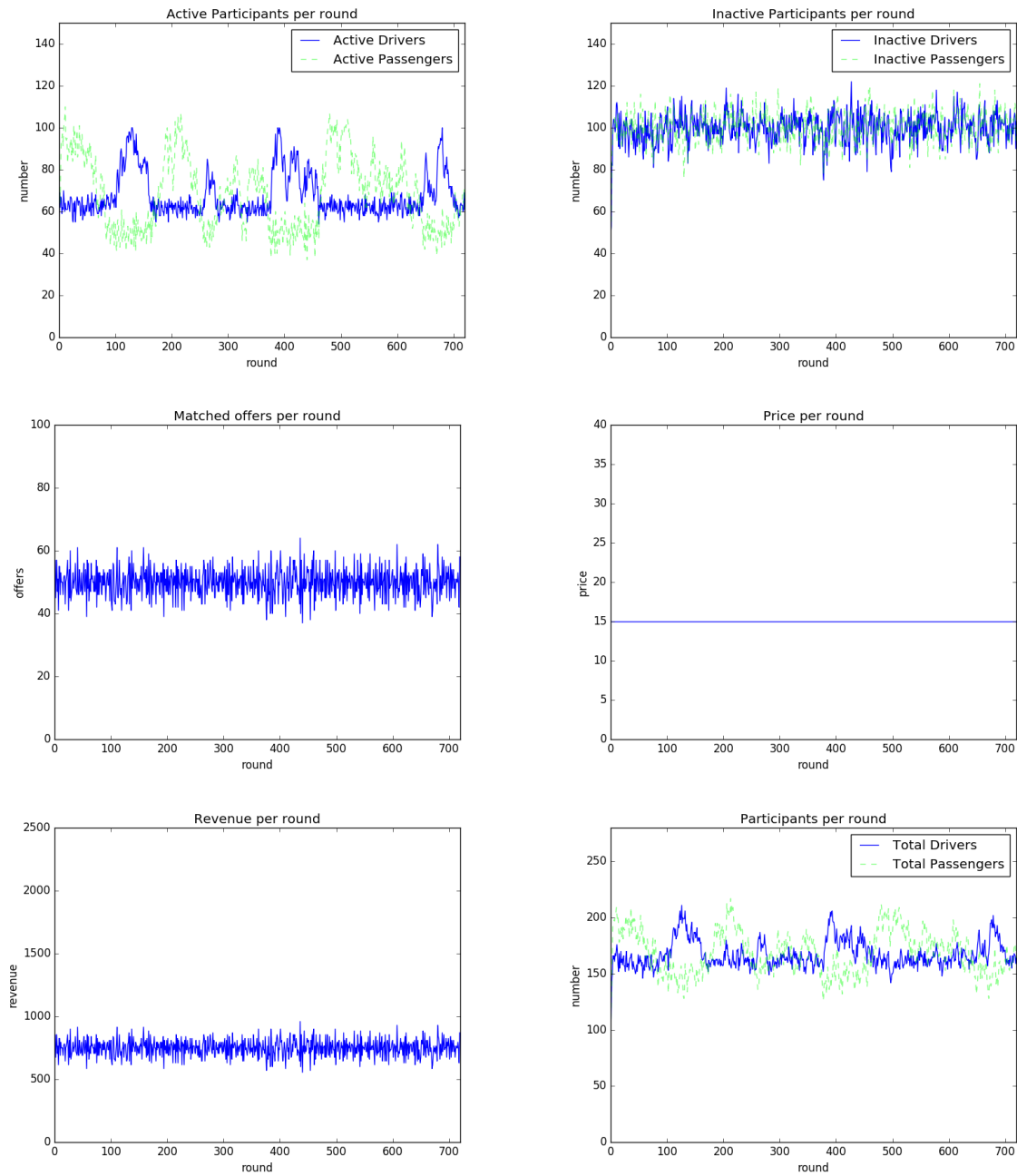


Figure 5.2: Simulated long term reactions to our proportional pricing model

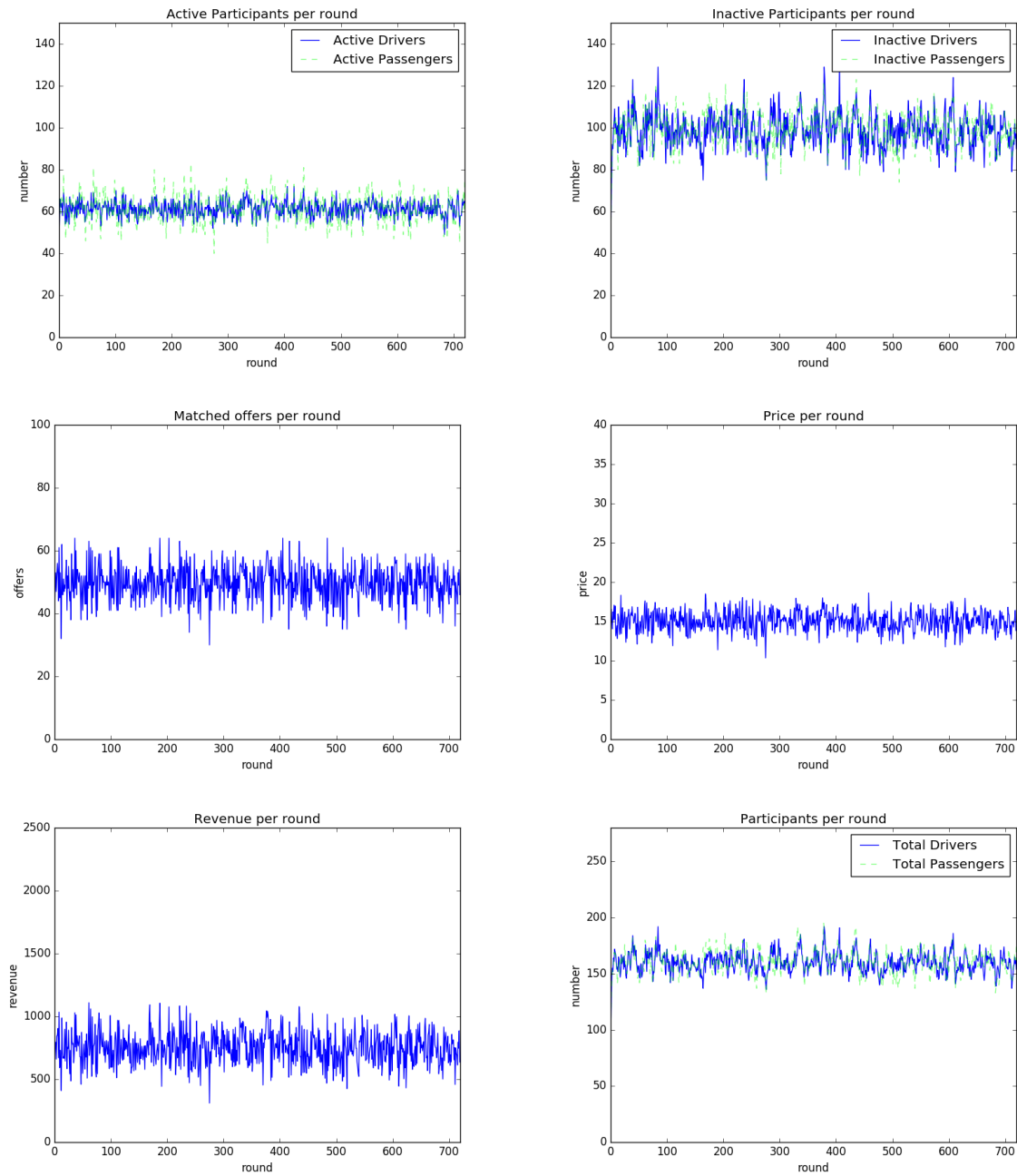




Figure 5.3: Simulated long term results for our batch update pricing model

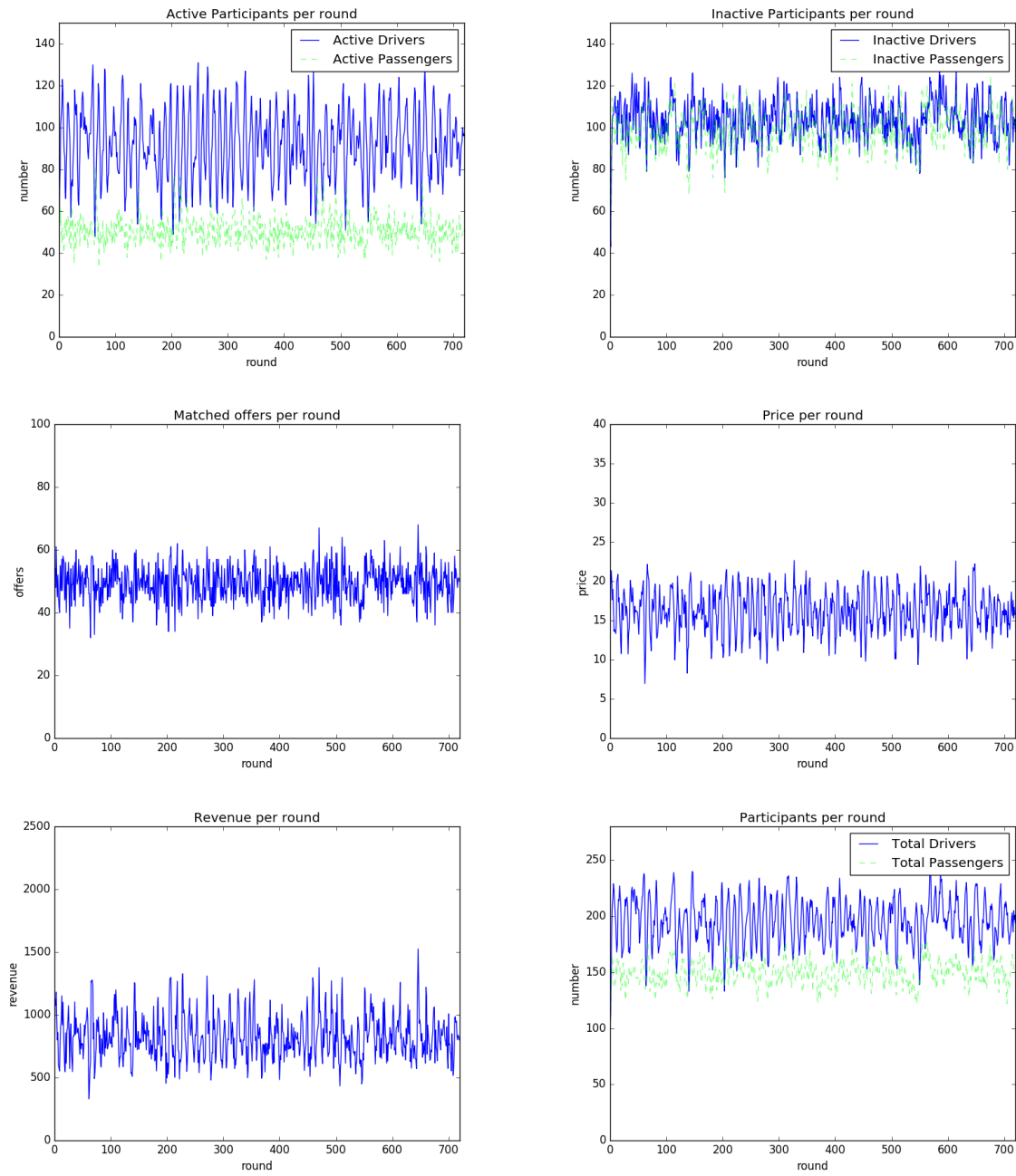


Figure 5.4: Simulated long term results for our reinforcement learning pricing model

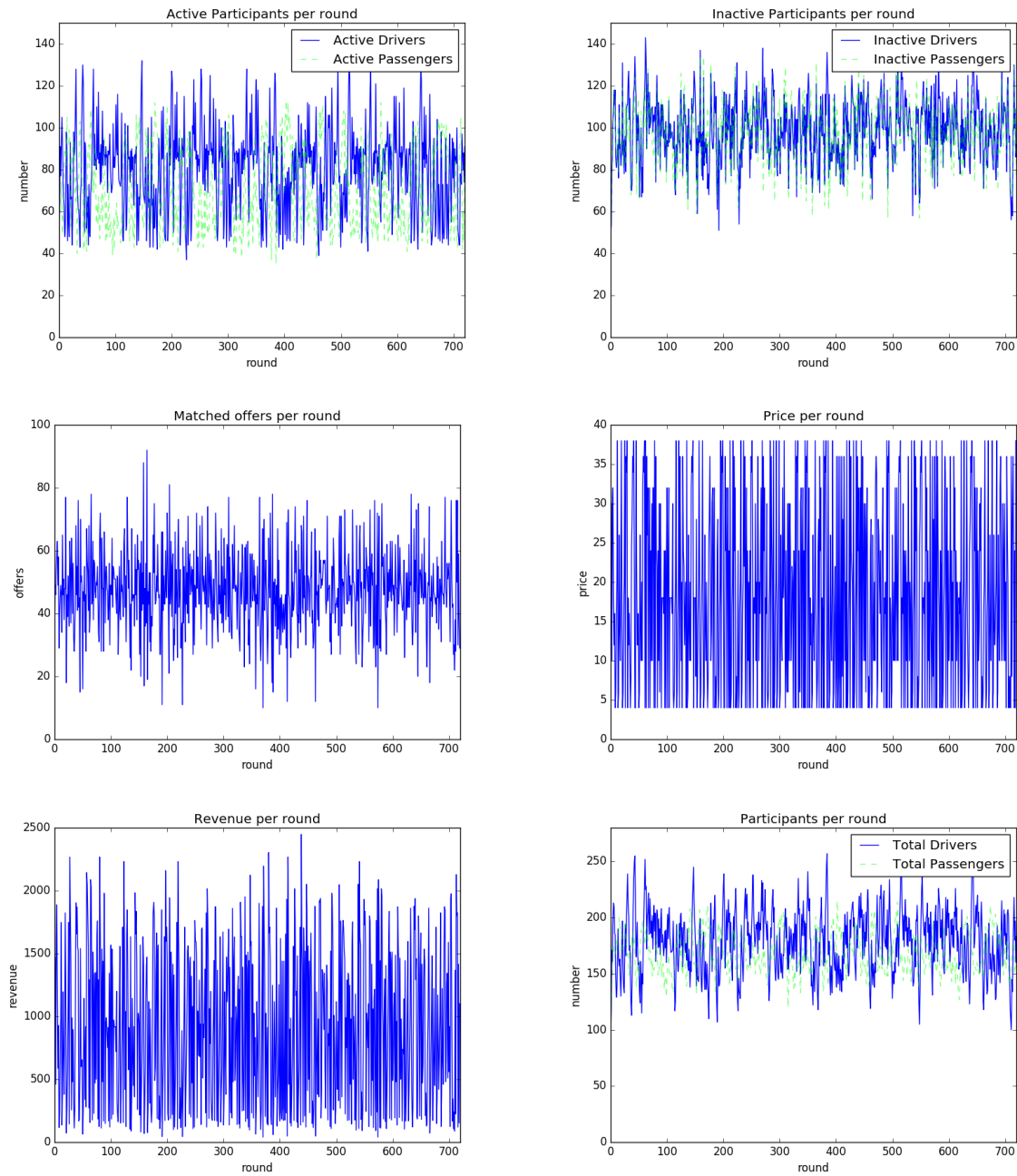
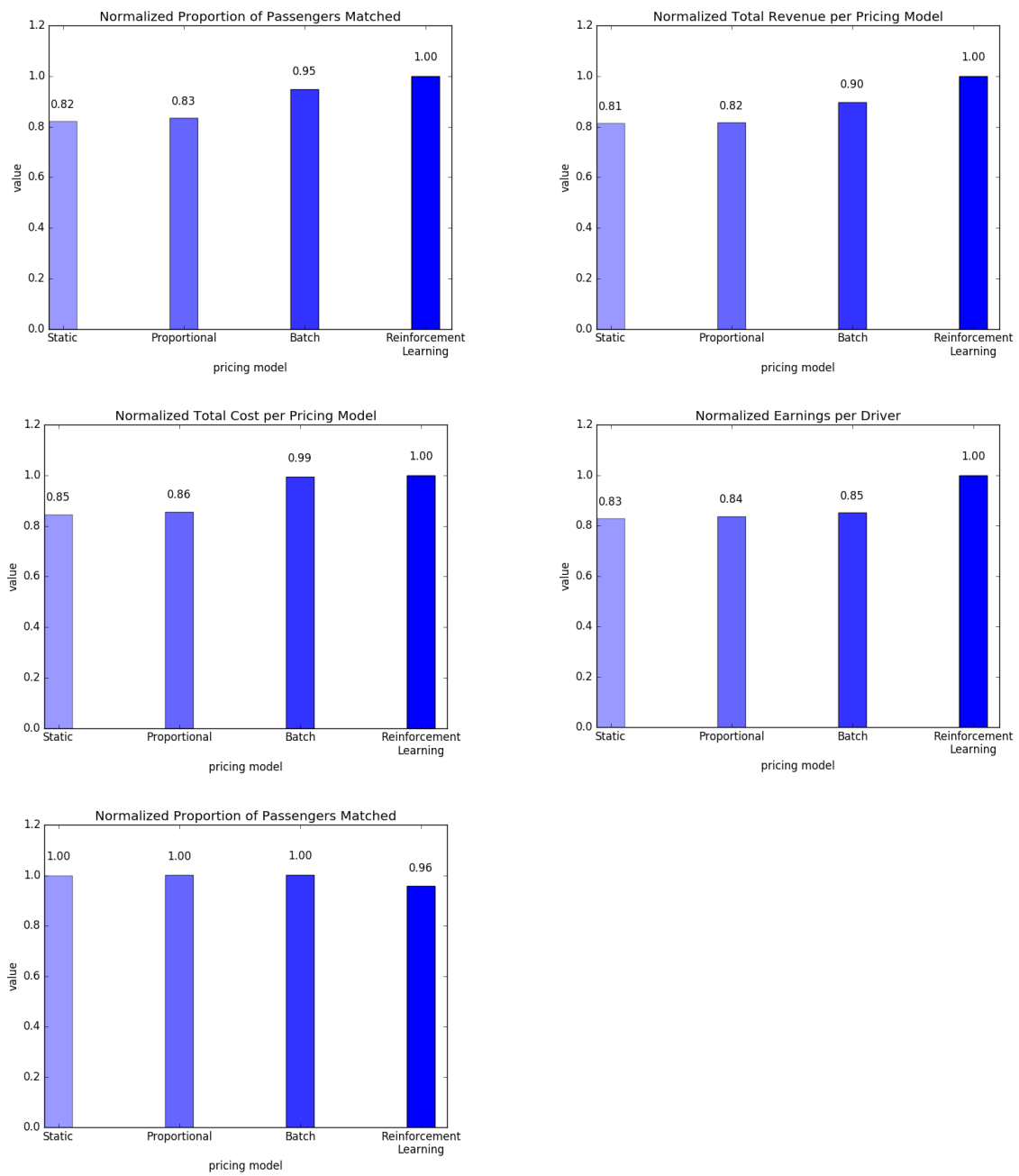


Figure 5.5: Comparison of pricing models in our long term simulation



# Chapter 6

## Future Work

### Building a More Robust Simulation

The simulation we are currently using is quite simple. The state space is small in a sense that it is only a combination of number of passengers, number of drivers, and price. The main limitation is that it has no geographical data and can only model one lane. We can add geographical data for each driver and passenger agent in the simulation. This would allow us the simulation to model lanes and agents in a geographical area.

The pain-point for adding geographical data is that we have to add location into the state space and our agent preference models. For each agent, their preferences will now depend on their location relative to the lane location and the location of the other relevant agents in the environment. In addition, adding geographical data results in a huge state space. However, we have known methods to deal with large state spaces, such as function approximation. We can use either a linear combination of state features or a neural network as good function approximators for this problem. We could also approximate lane similarity by using the cosine distance between two lane vectors.

### Inverse Reinforcement Learning

In our current simulation, we are simulating agent preferences based on market research and what we intuitively think how humans should act. However, if we are able to obtain a large dataset of real market data, we can apply inverse reinforcement learning techniques to learn actual agent preferences. With these learned preferences, our simulated agents should be able to act more similarly to real people.

### Multi-Competitor Landscape

Currently we model losing an agent as a fixed cost per agent lost. It would be interesting to see how our pricing agent acts in the presence of other competitors in the marketplace. A simple way of simulating this would be to pit two or more of our pricing agents against

each other in the same simulation and see how they react. We might be able to see more interesting behaviour and tactics surrounding agent retention.

## **Modeling Natural Growth**

One factor we have not modeled is natural growth of our agent population. We only consider active customer acquisition in our current simulation. This creates a pessimistic prediction of customer and user growth. In real life, our agent population may grow due to word of mouth, peer recommendations, and natural discovery. These growth factors need additional market research before we can determine how to accurately apply them to the simulation.

## **Real-World Application**

At the time of writing this paper, I am working as the cofounder of Dray Technologies Inc. Dray is an online and mobile marketplace for the on-demand trucking industry. The simulation created here can be easily applied to the spot brokering aspect of trucking. I plan to apply the reinforcement learning techniques used in this paper to generate instant prices for Dray's trucking shipments. We can also use real market data obtained from our shipments to create better simulations. It will be interesting to see how an automated agent's predicted prices compare to marketplace rates.

# Chapter 7

## Conclusion

In this paper, we built a simulation that models agent behaviour in an on-demand marketplace. We experimented with different pricing models in both short and long term simulations. We have shown that using reinforcement learning is an applicable solution to this problem. Overall, our reinforcement learning agent does the best job with regards to revenue and profit maximization. It beats out other pricing agents in both short term and long term simulations. The RL agent also performs the best when factoring in earnings per driver. However, the RL agent seems to maximize profits at the expense of ensuring matches for passengers. With these results in mind, a reinforcement learning agent can easily be expanded to more complex state spaces and simulations.

The main motivation of writing this paper is to explore and better understand the pricing problems that many on-demand companies face today. It is no longer sufficient to just forecast market uncertainties, because companies need a way to react to changes in real-time. An automated pricing agent is able to ingest data with scale and frequency while maximizing profits without human supervision. In practice, a company using a reinforcement learning automated pricing agent would be able to streamline its operations and improve its bottom line while retaining drivers and riders.

# Bibliography

- [1] *2010-2013 New York City Taxi Data*. 2016. URL: <https://publish.illinois.edu/dbwork/open-data/>.
- [2] Lantseva et al. “Data-driven Modeling of Airlines Pricing”. In: *Procedia Computer Science* 66 (2015), pp. 267–276.
- [3] Malighetti et al. “Pricing strategies of low-cost airlines: The Ryanair case study”. In: *Journal of Air Transport Management* 145 (2009), pp. 195–203.
- [4] Raju et al. “Reinforcement Learning Applications in Dynamic Pricing of Retail Markets”. In: *E-Commerce, 2003. CEC 2003. IEEE International Conference on* (2003).
- [5] Sousa et al. “Urban Logistics Integrated in a Multimodal Mobility System”. In: *IEEE Intelligent Transportation Systems* (2015).
- [6] Yossi Aviv and Amit Pazgal. “A partially observed Markov decision process for dynamic pricing”. In: *Journal of Consumer Research* 51 (9 2005), pp. 1400–1416.
- [7] Dimitris Bertsimas and Georgia Perakis. “Dynamic Pricing; A Learning Approach”. In: *Mathematical and computational models for congestion charging* (2006), pp. 45–79.
- [8] Ruth N Bolton. “A dynamic model of the duration of the customer’s relationship with a continuous service provider: The role of satisfaction”. In: *Marketing science* 17 (1 1998), pp. 25–26.
- [9] Martin. Christopher. “The Agile Supply Chain: Competing in Volatile Markets”. In: *Industrial Marketing Management* 29 (2000), pp. 37–44.
- [10] Pascal Courty. “Pricing Challenges in the Live Events industry: A Tale of Two Industries”. In: *IMC Roosevelt University* (2015).
- [11] Joan Morris DiMicco, Pattie Maes, and Amy Greenwald. “Learning curve: A simulation-based approach to dynamic pricing”. In: *Electronic Commerce Research* 3 (3 2003), pp. 245–276.
- [12] *Disney Introduces Demand-Based Pricing at Theme Parks*. 2016. URL: <http://www.nytimes.com/2016/02/28/business/disney-introduces-demand-based-pricing-at-theme-parks.html> (visited on 02/27/2016).

- [13] Benjamin Edelman Edelman, Michael Ostrovsky, and Michael Schwarz. “Internet Advertising and the Generalized Second Price Auction: Selling Billions of Dollars Worth of Keywords”. In: *American Economic Review, American Economic Association* 97 (2001), pp. 242–259.
- [14] Eyal Even-Dar and Yishay Mansour. “Learning Rates for Q-learning”. In: *Journal of Machine Learning Research* 5 (2003), pp. 1–25.
- [15] Marshall Fisher et al. “Making Supply Meet Demand”. In: *Harvard Business Review* (2015).
- [16] Gallego Guillermo and Garrett Van Ryzin. “Optimal dynamic pricing of inventories with stochastic demand over finite horizons”. In: *Management science* 40 (8 1994), 999=1020.
- [17] Jonathan Hall, Cory Kendrick, and Chris Nosko. “The Effects of Ubers Surge Pricing: A Case Study”. In: *The University of Chicago Booth School of Business* (2015).
- [18] Kelly L. Haws and William O. Bearden. “Dynamic pricing and consumer fairness perceptions”. In: *Journal of Consumer Research* 33 (3 2006), pp. 304–311.
- [19] Anton Leendert Hempenius. “Monopoly with random demand”. In: *Rotterdam University Press* (1970).
- [20] Justin Hienz. “Defining the Data Movement”. In: *The Future of Data-Driven Innovation* (2015).
- [21] *How Uber Chooses Its Surge Price Cap in Emergencies*. 2016. URL: <http://abcnews.go.com/Business/uber-chooses-surge-price-cap-emergencies/story?id=28494303>.
- [22] Junling Hu and Michael P. Wellman. “Multiagent reinforcement learning: theoretical framework and an algorithm”. In: *ICML 98* (1998).
- [23] Zhu Ji and KJ Ray Liu. “Multi-stage pricing game for collusion-resistant dynamic spectrum allocation”. In: *Selected Areas in Communications, IEEE Journal* 26 (1 2008), pp. 182–191.
- [24] Jeffrey O Kephart, James E. Hanson, and Amy R. “Dynamic pricing by software agents”. In: *Computer Networks* 32 (6 2000), pp. 721–752.
- [25] Jordan Kobritz and Steven Palmer. “Dynamic Pricing: The Next Frontier in the Evolution of Ticket Pricing in Sports”. In: *International Handbook of Academic Research and Training* 10 (2010).
- [26] *Lyft SF Driver Bonus*. 2016. URL: [www.lyft.com/SFBonus](http://www.lyft.com/SFBonus).
- [27] Aranyak Mehta et al. “AdWords and generalized online matching”. In: *Journal of the ACM* (2007), p. 22.
- [28] *NY Taxicab Rate of Fare*. 2016. URL: [http://www.nyc.gov/html/tlc/html/passenger/taxicab\\_rate.shtml](http://www.nyc.gov/html/tlc/html/passenger/taxicab_rate.shtml) (visited on 03/30/2016).



- [29] Edwin Pednault, Naoki Abe Abe, and Bianca Zadrozny. “Sequential Cost-Sensitive Decision Making with Reinforcement Learning”. In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (2002).
- [30] C. V. L. Raju, Y. Narahari, and K. Ravikumar. “Learning dynamic prices in electronic retail markets with customer segmentation”. In: *Annals of Operations Research* 143 (1 2006), pp. 59–75.
- [31] Jimmy Ssu-Ging Shih. “Applying Congestion Pricing at Access Points for Voice and Data Traffic”. PhD thesis. EECS Department, University of California, Berkeley, 2003. URL: <http://www.eecs.berkeley.edu/Pubs/TechRpts/2003/5572.html>.
- [32] Terry Taylor. “On-Demand Service Platforms”. In: *Social Science Research Network* (2016).
- [33] Gunnar T Thowsen. “A dynamic, nonstationary inventory problem for a price/quantity setting firms”. In: *Naval Research Logistics Quarterly* 22 (3 1975), pp. 461–476.
- [34] *Uber Promo*. 2016. URL: <https://www.uber.com/promo/>.
- [35] A. S. De Vany and T. R. Saving. “The Effects of Ubers Surge Pricing: A Case Study”. In: *The American Economic Review* 67 (2015), pp. 583–594.
- [36] Edward Zabel. “Multiperiod monopoly under uncertainty”. In: *Journal of Economic Theory* 5 (3 1972), pp. 524–536.