Platform Pricing Algorithms: Examples, Fundamental Challenges, Potential Solutions

Kanishka Misra,* Kenneth C. Wilbur†

December 12, 2024

Abstract

We consider platform pricing algorithms, i.e., pricing algorithms to learn optimal platform prices to customers. We review four papers that carefully describe platform pricing algorithms, finding that none of them internalize the network externalities that fundamentally define platform businesses. Next, we offer a modeling framework to consider platform pricing algorithms. We make 6 remarks on the substantial difficulties involved in designing platform pricing algorithms and illustrate them using simulations. Finally, we make four suggestions to feasibly design platform pricing algorithms.

Keywords: Algorithmic Pricing, Platforms, Platform Pricing Algorithms

1 Introduction

Algorithmic pricing and multi-sided platforms are prominent features of the digital economy. Firms increasingly use algorithms and machines to automate pricing decisions. Multi-sided platforms organize increasingly large proportions of economic activity, to the point that all five of the largest corporations by market value in October 2024–Apple, Nvidia, Microsoft, Alphabet and Amazon–are platform businesses.

*Boston College, bc.edu/kanishka-misra.html

†University of California, San Diego, kennethcwilbur.github.io/website/

The intersection is also populated: Platform businesses employ pricing algorithms. Therefore, one might reasonably expect a large academic literature to study "Platform Pricing Algorithms" (PPA), i.e. pricing algorithms that platforms use to set customer prices, accounting for the network externalities that fundamentally define the platforms themselves. Yet there are few academic studies of platform pricing algorithms (Sanchez-Cartas and Katsamakas, 2024). The divergence is a puzzle.

In fact, some platform prices change rather infrequently. For example, Amazon has only revised its U.S. Prime subscription service price three times since its launch in 2005 (Brown, 2023). The price invariance stands despite numerous service quality changes, such as the launch of Prime Video in 2011, Prime Reading in 2016, and the introduction of advertisements in Prime Video programs in 2024. Similarly, Amazon has left its individual seller transaction fee schedule unchanged since at least 2010 (Holden, 2010). Such price constancy does not suggest frequent optimization.

In this chapter, we selectively review literature on platform pricing algorithms. We survey four sophisticated pricing algorithms implemented by Walmart, Amazon, ZipRecruiter and AirBnB: all of the available approaches we can find that are described carefully in published documents. None of these four pricing approaches internalize direct or indirect network externalities, which might be viewed as surprising given these firms' sophistication and resources.

Next, we develop a framework for thinking about platform pricing algorithms, by considering algorithmic pricing approaches within a general platform model with heterogeneous network externalities. We remark upon several fundamental challenges to help explain why platform pricing algorithms have been rarely studied or implemented. We illustrate some of the challenges using simulations. Finally, we offer informed speculation about approaches to make progress toward overcoming these fundamental challenges.

2 Pricing Algorithms Implemented by Platforms

We found four platform pricing algorithms that are described carefully in published documents, by Walmart (section 2.1), Amazon (2.2), Ziprecruiter (2.3) and AirBnB (2.4). Each subsection explains platform pricing algorithm details, advantages and challenges.

2.1 Walmart

Ganti et al. (2018) describes pricing algorithms that Walmart deployed as a seller in its own online marketplace. The study includes a baseline passive learning process in which prices were optimized given prior demand estimates. It also proposes an active learning algorithm in which a Bayesian estimator is used to update demand estimates, then a greedy algorithm actively explores the space of possible prices.

More specifically, the authors describe a baseline frequentist passive-learning algorithm and their proposed Bayesian active-learning algorithm. Both algorithms condition on an underlying constant-elasticity parametric demand function, and a "black-box" demand forecasting system in which sales are predicted as a function of prices. In the passive-learning algorithm, past price changes are used to estimate each product's own-price demand elasticity parameter, which is then used to calculate each product's predicted revenue-maximizing price. The authors explain that this approach does not actively explore the space of possible prices, resulting in relatively invariant prices, which in turn limits the number of prices changes available to accurately estimate price elasticities.

Authors design their active-learning replacement by placing a Gaussian prior on each product's elasticity parameter, which they show leads to a tractable posterior distribution for each product's revenue-maximizing price. They then sample from the posterior distributions of elasticity parameters, reject and resample any positive elasticity parameters sampled, and then update prices based on the posterior distributions. They show via simulations and an implementation that their approach leads to higher revenues in a non randomly selected basket of products, relative to a non-selected basket in which the status-quo approach was applied.

We applaud the active-learning component of the price-setting algorithm. We also appreciate the application of microeconomic theory and the application of a popular demand function to optimal pricing. However, the pricing algorithm lacks some notable features. It rules out all non-constant elasticity demand functions by assumption; it foregoes accommodating any cross-product substitution within its algorithm (the only pricing parameter is an own price elasticity); it should have been tested against a randomized control group of non-competing products; and it does not incorporate algorithmic effects on buyer or seller platform trial, repatronage, purchases, sales, or

expenditures. It appears difficult to internalize customer network externalities within the bounds of this pricing algorithm.

2.2 Amazon

Amazon operates a large online retail marketplace, and also participates as the largest seller in that marketplace, as well as offering other related services like inventory management, storage, shipping, cloud computing and advertising. We focus on a specific pricing algorithm it offers to sellers in its marketplace, as it has published details. Note that we lack detailed information about its use of platform pricing algorithms in other capacities, such as its role as a seller in its own marketplace. Notably, Amazon promised in 2001 that it "never will test prices based on customer demographics." (Amazon, 2000)

Cooprider and Nassiri (2023) describe an Amazon "pricing experimentation platform" designed to help sellers discover optimal prices through experimentation. The approach seeks to meet three constraints. First, it avoids serving random prices to different customers at the same time, in strict keeping with the 2001 promise. Second, it acknowledges the empirically important fact that different products exhibit heterogeneous demand responses to price. Third, it acknowledges the empirical importance of competitive spillovers between products–due to business stealing by substitutes and demand expansion among complements–as a threat to the Stable Unit Treatment Value Assignment (SUTVA) assumption underlying experimental analysis.

The Amazon Pricing Labs price algorithm includes four main elements: (1) an initial pre-treatment period and multiple intra-test periods to measure demand trends in the absence of price manipulations; (2) clustering of similar products based on initial demand trends; (3) cluster randomization wherein clusters of products are assigned to price treatment and control conditions; and (4) switchback treatments (rather than fully time-randomized prices), in which clusters of treated products are switched between treatment and control. The specific price elasticity estimator relies on causal forests using a difference-in-differences logic, namely comparing randomly-assigned price levels within treated product clusters as one difference, and comparing randomly-assigned price levels across product clusters as the second difference. Standard errors are obtained via bootstrap.

On the positive side, the Amazon Pricing Labs algorithm is a sophisticated experimental design and estimator which acknowledges multiple challenges within a marketplace environment. However, we see three particular limitations which prevent us from classifying it as a platformo pricing algorithm. First, it is not a fully automated pricing algorithm, in that it does not set prices in real time, it does not automatically respond to current changes in demand conditions, and it does not offer a state-specific pricing function. Second, it does not internalize the effect of individual product price optimization on shoppers' purchases, visit frequency or spending with the seller or in the marketplace. Third, it does not internalize the effects of individual product price optimization on sellers' continued marketplace usage, product line, profitability, advertising spending or other related strategic variables.

A feasible extension of the Cooprider and Nassiri (2023) framework might be to estimate an optimal price schedule as a function of current demand and inventory conditions, to enable sellers to algorithmically update their prices depending on the experimental findings while accounting for other shifting factors. However, as in the case of Walmart, it is not easy to see how this pricing approach could be modified to internalize customer network externalities.

2.3 ZipRecruiter

ZipRecruiter is a platform where firms can advertise job listings and job seekers can apply for jobs. The platform charges firms a fee to advertise jobs; jobseekers access job ads for free. Dubé and Misra (2023) consider ZipRecruiter's advertisement pricing decision. Prior to their analysis, all employers were charged $99 per job listing. The authors designed a field experiment in which firms were randomly exposed to 1 of 10 prices between $19 and $399. They estimated a flexible demand model accounting for numerous firm characteristics. Authors worry about overfitting so they use a weighted lasso model to maintain predictive accuracy.

Using demand estimates, authors suggested a large increase in the uniform price of ZipRecruiter, from $99 to $249. Further this model can be used to calculate an optimal price discrimination schedule, in which listing prices depend on firm characteristics. ZipRecruiter implemented the algorithm in a second field experiment, and the authors estimated that the pricing algorithm increased revenue by 86%.

This paper provides a powerful implementation of price discrimination but it also highlights an important concern with a purely algorithmic approach. The authors explain that "a concern with our personalization scenario is that about one quarter of our recommended prices exceed the highest price [tested] in the experiment, $399, with many in excess of $1,000. ZipRecruiter's management team indicated that they would be unlikely to consider prices above $499." Maximum prices are often a tuning factor in pricing algorithms.

This paper offers strong evidence that ZipRecruiter's platform pricing algorithm increased its short-term revenue substantially. However, like the previous platform pricing algorithms we have discussed, the proposed pricing algorithm did not internalize its effects on the other side of the market (jobseekers). One might speculate that increasing listing prices could change the number of listings on the platform. We might further speculate that fewer job ads due to higher ad prices could make jobsearching more efficient for high-ability jobseekers by eliminating marginal job ads, and simultaneously make jobsearching less likely to produce matches for less-able jobseekers.

In fact, the paper reports a baseline conversion of 23% under the status quo $99 pricing policy, which then decreased to 15% under the uniform-price of $249 as well as under price discrimination based on firm characteristics. Simultaneously, the number of job postings decreased by 35%. That might increase the platform's utility to well qualified jobseekers, but it also may reduce the platform's overall utility to more marginal jobseekers by crowding out some listings.

It would be interesting to consider how many jobseekers may have switched platforms or begun multihoming as a result of the new listing pricing policy. It also would be interesting to consider how the new pricing policy affected paid jobseeker services on the platform, or whether it increased platform utility to employers by increasing competition among jobseekers. Finally, one may reasonably speculate that salary negotiations may have changed as a result of the job-ad price increase, as the number of respondents to each job ad may have changed in equilibrium. As before, it is not easy to see how to extend the proposed pricing algorithm to internalize direct or indirect network externalities.

2.4 AirBnB

AirBnB is a platform that connects property hosts with guests seeking short-term rentals. On AirBnB, hosts set future reservation prices for their properties. The price a host charges is a transaction fee paid by the guest, with AirBnB charging commission fees to both sides. In the current structure hosts are charged 3% and guests are charged 14.2% of booking price.[1]

Hotel reservation pricing is a high-dimensional, complex problem (Cho et al., 2018; Joo et al., 2020), as the pricing manager has to price substitutable rooms at numerous future dates and booking times, subject to capacity constraints. AirBnB hosts face a similarly complex optimization problem, especially for those that offer multiple proximate properties, such as distinct rooms within the same property. AirBnB piloted a pricing algorithm to assist hosts with price recommendations. Algorithm details are published in Ye et al. (2018).

The first step of the algorithm is a gradient boosting machine learning model. The model is trained to predict time-varying booking probabilities for each property for each future day as a function of property features, time features, and local dynamics for hosts and guests. The authors point out a number of challenges of this prediction problem; for example, there may be insufficient past price variation within individual listings, so it can be hard to separate price effects from unobserved listing attributes. The implemented solution is a market-specific prediction model accounting for the geographic density of listings. This is a challenging prediction problem for the platform, even though it has thick market data. For hosts, who lack such data, the demand prediction problem is even more challenging as they cannot easily observe market shocks or how they propagate across properties and future time periods.

A standard pricing algorithm would maximize expected profits based on the calibrated prediction model. Ye et al. (2018) argue this is not useful to hosts due to the large errors in the prediction model. Instead, they consider a "fitting model," where they fit a flexible parametric pricing function that relates the prediction output to the observed prices charged in the data. In particular, they specify a loss function where if a listing was booked (not booked) at price p, the optimal price must be bounded above (below) the price observed in the data p. They estimate fitting models for each individual listing rather than aggregating across listings. They use market level price recommendations to solve the cold-start problem for a new listing.

[1] Retrieved from AirBnB.com

An advantage of AirBnB's platform pricing algorithm does provide hosts a simple tool for exploring dynamic pricing, i.e. setting many future prices depending on expected future demand conditions. This can be particularly useful for hosts who have not adopted dynamic pricing due to imperfect information (Huang, 2022), for example less experienced hosts.

A downside of the smart pricing algorithm is that it does not account for hosts' costs: it maximizes expected revenues rather than expected profits. Huang (2022) concludes "if the platform were to enforce this algorithm, prices would have been much lower than the market-clearing level, and sellers would have been worse off." Host commentary[2] and Zhang et al. (2021) corroborate that hosts who adopt AirBnB's pricing algorithm tend to have lower costs and higher occupancy rates. AirBnB platform prices condition on completed transactions, so recommending suboptimal prices to hosts may improve platform profits by increasing total sales. Indeed Zervas et al. (2017) show that AirBnB competed for hotel traffic mainly from lower-priced budget hotels. These misaligned incentives further show the potential complexity of platform pricing.

Another drawback of the pricing algorithm is that it does not internalize or price the effect of keeping guests and hosts on the platform. This is a nontrivial challenge given that it might either require a different algorithmic approach, or that each individual host's price recommendation might condition on each guest's unobserved counterfactual reservation strategy in the case that an encountered price is too high.

2.5 Other pricing algorithms used by platforms

Uber is a ride-hailing platform that connects riders with drivers. The platform sets the price for each ride and drivers are paid a commission, however the pricing algorithm is not communicated transparently. Uber does say that "Upfront fares are calculated based on current driving conditions, like destination and demand," which raises the possibility that its pricing internalizes some network externalities, such as local rider and driver platform usage. [3] As an example Castillo (2023) shows evidence for surge pricing by considering demand and supply in different areas of Houston. While the specific details of Uber's pricing algorithm are not public, the academic literature describes potential algorithms that capture similar features (e.g., Garg and Nazerzadeh, 2022; Yan et al.,

[2] https://www.bnbcalc.com/reviews/AirBnB-smart-pricing-review

[3] https://www.uber.com/us/en/drive/how-much-drivers-make/

2020). It may be that Uber is operating a platform pricing algorithm (i.e., pricing to internalize some network externalities) without disclosing the details.

Much is known about how advertising platforms like Google, Meta and others algorithmically sell inventory (e.g., Varian, 2007; Wilbur et al., 2013). However, such algorithms often employ auctions rather than posting prices. Auctions elicit willingness to pay schedules from advertisers as an input to the pricing process, whereas pricing algorithms set take-it-or-leave-it prices that customers may take or leave without indicating willingness to pay.

3 Platform Pricing Algorithms: Fundamental Challenges

First we specify a reasonably general model of heterogeneous platform customers. Then we specify a reasonably general model of algorithmic pricing. Then we remark on some of the fundamental challenges of implementing platform pricing algorithms–that is, applying algorithmic pricing techniques without a setting of heterogeneous platform customers. We provide simulation evidence to illustrate these remarks and offer informed speculation about possible collusive outcomes.

3.1 A model of heterogeneous platform customers

We specify a flexible model of platform customers that generalizes canonical models, to help clarify and explain our claims below. We do not specify the model for the purpose of solving or estimating it, but rather to generally represent the heterogeneous platform customer behaviors that define the environment in which a platform pricing algorithm would seek to learn optimal prices. Customer heterogeneity is a first-order issue in platform settings, as top sellers/creators often enable exponentially more interactions than modal sellers/creators.

We consider two types of agents which we refer to as buyers (b) and sellers (s), but most of our ideas extend to other common settings, such as drivers/riders, hosts/guests, advertisers/creators/-consumers, etc. We index individual agents (of any type) with i, agents of the opposite type with j, and the two agent types with $k \in \{b, s\}$. We indicate platform usage by individuals of the same and other type with v_i and v_j. We consider a monopoly platform that charges access fees, but we believe that most remarks extend to other pricing models and competitive settings.

Assume user i of type k has platform usage utility of

$$U_i^k = \omega_i^k - \alpha_i^k \cdot p^k + \sum_{\forall i' \neq i} \beta_{ii'}^k \cdot v_{i'} + \sum_{\forall j} \gamma_{ij}^k \cdot v_j \tag{1}$$

where:

- ω_i^k is the platform usage utility for user i of type k, without price and network externalities;
- α_i^k is price sensitivity of user i and p^k is the price to type k;
- Direct network externalities: $v_{i'}$ indicates platform usage by same-type agent i', from which agent i experiences externality $\beta_{ii'}^k$, with externalities summed over all agents of the same type i'. Direct network externalities may be either positive when agents collaborate in some way, for example in a multiplayer online game or when sharing product information in reviews; or negative, for example when same-type agents compete for scarce resources and matches, such as in product or dating markets. We presume heterogeneous agent platform usage will generate heterogeneous direct network externalities as agents often differ greatly in their attributes, prominence and spillovers toward others. For example, creator followings in social media vary enormously due to heterogeneous creator attractiveness, posting quality and posting frequency; seller transactions vary enormously within transaction marketplaces; and app usage varies enormously within app stores.
- Indirect network externalities: v_j indicates platform usage by other-type agent j, from which agent i experiences externality γ_{ij}^k, with the externalities summed over all agents of the other type k. Indirect network externalities may be either positive when agents collaborate in some way, for example in a credit card network in which merchants and consumers transact; or negative, for example in attention-economy platforms where consumers may prefer to avoid some advertiser messages. Again, we presume heterogeneous agent platform usage will generate heterogeneous indirect network externalities.

User utility is zero without platform use. Total platform usage by agent type k is $n^k(p^b, p^s)$. The platform chooses prices (p^b, p^s) to maximize profits $\Pi = p^b \cdot n^b(p^b, p^s) + p^s \cdot n^s(p^b, p^s)$ then observes which users pay for access. Prices may be negative, for example in credit cards where cardholders may be subsidized with rewards.

3.2 Algorithmic pricing frameworks

Businesses have used algorithmic pricing for generations. A pricing algorithm is simply a rule-based approach to setting prices. However, digitization has led most researchers to define algorithmic pricing more narrowly (Smith and Tadelis, 2021). Most often, the term refers to computerized pricing rules which try to maximize an objective function (usually, profits) within feasible constraints, based on empirical feedback and frequent reconsideration without human intervention. The prices being set may include piece rates, two- or three-part tariffs, or other related fees; and may discriminate across markets, segments or consumers.

The objective function of a learning algorithm is to experiment with feasible prices (learn) while maximizing the possible profit for the firm (earn). The algorithm considers profit as an unknown function of prices (more generally, actions) to realized profits (more generally, outcomes). Mathematically, $\pi : (p^b, p^s, \sigma) \to \mathbb{R}$, where σ represents the current state of the world. Rather than trying to learn the full profit function the algorithm focuses on sampling pricing actions in hopes of moving toward profit maximizing prices. The fundamental complication of platform pricing algorithms is that agent responses are not fully individualistic, as direct or indirect network externalities such as participation or competition externalities change respondent payoffs, complicating optimal price learning.

There are three common learning frameworks within algorithmic pricing.

1. Suppose the firm has prior information about the payoff function π, e.g., historical demand or expertise. Then it may characterize π within a parametric demand model, possibly with priors on the unknown parameters; then use an unsupervised learning algorithm to estimate the parameters of the demand system while optimizing prices to maximize payoffs, as in Ganti et al. (2018).
2. Suppose the profit function is ambiguous but stable across pricing actions. In other words, we do not impose assumptions about the mathematical shape or structure of demand, but we assume the payoff function is likely to persist independently across sampled actions. Suppose the firm only knows the range of possible prices it can charge, say ranging between cost and some upper bound. The firm might simply use a multi-armed bandit learning algorithm, which experiments from available prices to maximize expected profits given what it

has learned to date. Or, to be even more effective, the firm might use the indexed bandit algorithm of Misra et al. (2019), which extended prior bandit algorithms to maximize expected payoffs under the assumption that demand does not slope upwards. The indexed bandit was shown to be asymptotically optimal for any weakly downward-sloping demand curve.

3. Finally, suppose the profit function is both ambiguous and potentially unstable, but we can measure some determinants of the function such as competitor prices or other market conditions. Then the firm might encode relevant conditions in σ and experiment over actions to find the payoff-optimal action for each σ; see, e.g., the Q-learning approach in Calvano et al. (2020). With sufficient sampling, this algorithm is guaranteed to converge to the true profit maximizing price under certain regularity conditions (Watkins and Dayan, 1992).

3.3 Remarks on Platform Pricing Algorithms

Applying these learning frameworks in a platform setting requires accounting for user interdependence of platform usage, wherein network externalities between users and between sides affect agents' price taking decisions. A very simple way to specify the state vector is $\sigma \in \{n^b, n^s\}$. A more precise way to specify the state vector would be $\sigma \in \{v^b, v^s\}$ wherein each v^k is a vector whose contents indicate platform adoption by each individual agent of type k. The former approach may be appropriate for settings in which only agent ratios or minimum numbers of agents suffice to account for network effects. The latter formulation is more complete and better equipped to handle extensive heterogeneity in agents' network externalities, at the cost of an exponentially larger state space. There also could be hybrid approaches between the two extremes in which we group similar agents within some discrete number of classes indexed by k, or project agent participation statistics into a lower-dimensional space based on continuous agent attributes.

Remark 1. Platform pricing algorithms require state-based learning frameworks.

The interdependence of agent participation decisions requires that platform pricing algorithms condition on the network externalities between customers and customer types. We show this by simulating a parsimonious setting in section 3.4.

Remark 2. Platform pricing algorithms need to explore exponentially larger state spaces than otherwise-similar non-platform pricing algorithms.

The additional complexity in platform settings requires additional sampling and time for convergence. Among non-parametric solutions, the statistical regret or learning cost from index-based methods such as multi-armed bandit reduces in $O(log(t))$ time (Auer, 2002), whereas Q-learning converges significantly more slowly with regret of $O(\sqrt{t})$. In both cases, the learning cost scales with the number of states (Kearns and Singh, 1998). Therefore in a platform application this could require a very large number of pricing experiments to reach optimality, even in the simple case where $\sigma = \{n^b, n^s\}$. High learning costs could make platform pricing algorithms intractable if state spaces and action spaces are not specified thoughtfully.

Remark 3. Platform pricing algorithms need to learn action interactions in order to find optimal actions.

Network externalities in equation 1 make it clear that agents' decisions are intertwined, so each time a pricing algorithm experiments on p^b it will change the returns to p^s and vice versa. And, when one price type p^k is far above the optimum, it may flatten the profit function and therefore make it more difficult to learn the other optimal price. As a limiting example, suppose p^b were so high that no buyers pay; then there may exist no p^s that attracts sellers to the platform.

Remark 4. Network externalities complicate price discrimination in platform pricing algorithms.

Consider platform pricing algorithms that seek to price discriminate across customers or customer types, e.g. the one-sided ZipRecruiter example of Dubé and Misra (2023). If we extend this framework to incorporate network effects on a platform we face a different challenge. Hajihashemi et al. (2022) study a theoretical model of price discrimination with network effects and show two implications for price learning/experiments: (a) price discrimination can decrease firm profits and (b) the impact of network effects will not be fully realized until later time periods. From a learning or experimental point of view this implies that consumer utility changes between when the pricing experiments were run and when the learnings were exploited. This is because some agents leave the market, therefore changing patterns of the network effects. Pricing algorithms therefore may need to consider multi-period profit realizations in order to evaluate pricing actions accurately

Remark 5. Enacting a platform pricing algorithm may change the nature of equilibrium between competing platforms, complicating the learning task.

A growing body of algorithmic pricing literature considers the role of competition between pricing

algorithms, as reviewed in Hansen et al. (2021) and Aparicio and Misra (2023). The literature shows conditions under which algorithms can converge to collusive (supra-competitive) equilibrium prices, or in which they may slide into super-competitive pricing games, and it shows that such algorithmic competition may switch between these two extremes rapidly.

The role of competition in platforms with consumer multi-homing has been discussed extensively in the theoretical literature (e.g., Bakos and Halaburda, 2020; Jullien et al., 2021; Teh et al., 2023). These papers highlight the tension between competition and pricing on each side the network. For example, Teh et al. (2023) show that the fee structure between buyers and sellers depends on the percentage of agents who multi-home.

The results in the current algorithmic collusion literature are limited to markets without network effects. The theoretical literature on platform markets shows that different equilibrium structures may occur depending on multi-homing behaviors by various types of agents. If high experimental prices induce some agents to multihome, or if low experimental prices render multihoming unnecessary, then the act of learning optimal prices may change the optimal prices themselves by changing the nature of competitive equilibrium between the competing platforms.

Remark 6. Network externalities among agents prevent experiments in the action space from producing unconfounded results.

We can randomize the prices, and we may learn something useful from the resulting agent decisions, but we cannot run valid experiments under general network externalities due to interdependence of agent payoffs. This is similar to well-known Stable Unit Treatment Value Assumption (SUTVA) violations that occur when researchers try to randomize treatments distributed by platforms (Eckles et al., 2017, 2018).

3.4 Simulation Evidence

We offer a parsimonious simulation to illustrate some of our remarks. We simulate the application of a pricing algorithm that disregards network effects to a data generating process with direct network effects among homogeneous agents of a single type. We assume the following.

- Rounds / time periods: A round is defined by 100 consumer arrivals making platform purchase decisions. Rounds are indexed by t, starting from round 0 with zero arrivals.

- Potential prices: We consider prices in \$0.05 intervals between \$0.05 and \$0.5. We assume zero marginal cost, so pricing is set to maximize revenue.

- Price update frequency: As in Hansen et al. (2021) and Calvano et al. (2020) all consumers face the same price within each round t. So, the price can change after each 100 consumer arrivals.

- Utility: Consumers have homogeneous preference parameters. Consumer i's utility in round t is given by

$$u_{i,t}^{buy} = \alpha + \beta p_t + \gamma s_{-i,t} + \varepsilon_{i,t}^{buy}$$
$$u_{i,t}^{nobuy} = \varepsilon_{i,t}^{nobuy}$$

  $s_{-i,t}$ is the share of other consumers who pay to access the platform within the same round t. Utility shocks ε_{it} are distributed type 1 extreme value, independently across consumers, rounds, and options. p_t is the platform price offered in round t. For specificity, we set the utility parameter values to $\alpha = -1$, $\beta = -3$ and $\gamma = 3$.

In this setting the optimal platform price is \$0.15, which may be verified by a grid search that incorporates knowledge of the data generating process. In our simulation we consider three non-parametric learning algorithms, none of which are endowed any information about the data generating process. First, we use the unconstrained bandit algorithm UCB1 (Auer, 2002) of Hansen et al. (2021). Second, we run an ε-greedy algorithm, a state-free version of the Q-learning algorithm in Calvano et al. (2020). The model excludes competition so there are no states in applying their algorithm. Third, we simulate a Thompson Sampling algorithm as used in the Walmart example of Ganti et al. (2018).

These algorithms are defined as follows:

1. Initialization: conduct 10 experiments in each price, ordered randomly.
2. Algorithm at round t
   - UCB1: for each price j calculate an index (UCB) as $UCB_{j,t} = \overline{\pi_{j,t}} + \sqrt{\frac{2log(t)}{n_{j,t}}}$ where $\overline{\pi_{j,t}}$ is the empirical mean of profits from charging price j in all prior rounds and $n_{j,t}$ is the number of rounds where price j has been charged before t.

Pick the price j with the highest index (UCB) in round t.

- ε-greedy: with probability ε_t experiment with a random price otherwise pick the price with the highest empirical mean profit $\overline{\pi_{j,t}}$. As in Calvano et al. (2020), we set $\varepsilon_t = e^{-\delta t}$ with $\delta = 5 * 10^{-6}$
- Thompson Sampling: As in Ganti et al. (2018), we assume the firm considers an aggregate demand model defined as $\frac{s_{j,t}}{1-s_{j,t}} = \alpha^{TS} + \beta^{TS} p_{j,t} + \xi_{j,t}$. This functional form is an aggregate demand model version of the logit demand system (Berry, 1994) and assumes an additional error term $\xi_{j,t}$ reflecting variation in choice shares across time due to small sample sizes. This model is estimated based on all data before round t. We draw from the asymptotic distribution of the estimated parameters and set the price in round r that maximizes profits given that draw.

We run all three pricing algorithms for 1,000,000 rounds and summarize the learned prices and profits in Figure 1. The left column of charts displays the results from UCB1, the middle column shows results of ε-greedy and the right column presents Thompson Sampling results. The top row shows the prices charged in each of the last 1,000 simulated rounds, and the bottom row displays the learned profit functions based on price; and two bar graphs showing true profits with network effects and true profits without network effects.

Figure 1 shows that all three algorithms converge to supra-optimal prices. The algorithms differ dramatically in price variation over the last 1,000 rounds. The three learned profit functions show very different curvatures. Finally, we notice that learned profit functions tend to be flatter at higher prices wherein direct network externalities are smaller. However, the high-price flattening varies across algorithms with the Thompson Sampling algorithm showing curvature that nearly matches the true data generating process among higher prices.

Finally, we ensure that our simulation code is correct and that the pricing biases are due to network effects. We re-run the same simulations by setting network externality parameter $\gamma = 0$, holding everything else the same, and show the results in Figure 2. This shows that all three pricing algorithms estimate an unbiased profit curve and learn optimal prices. Therefore, the biased prices are due solely to pricing algorithm misspecifications.

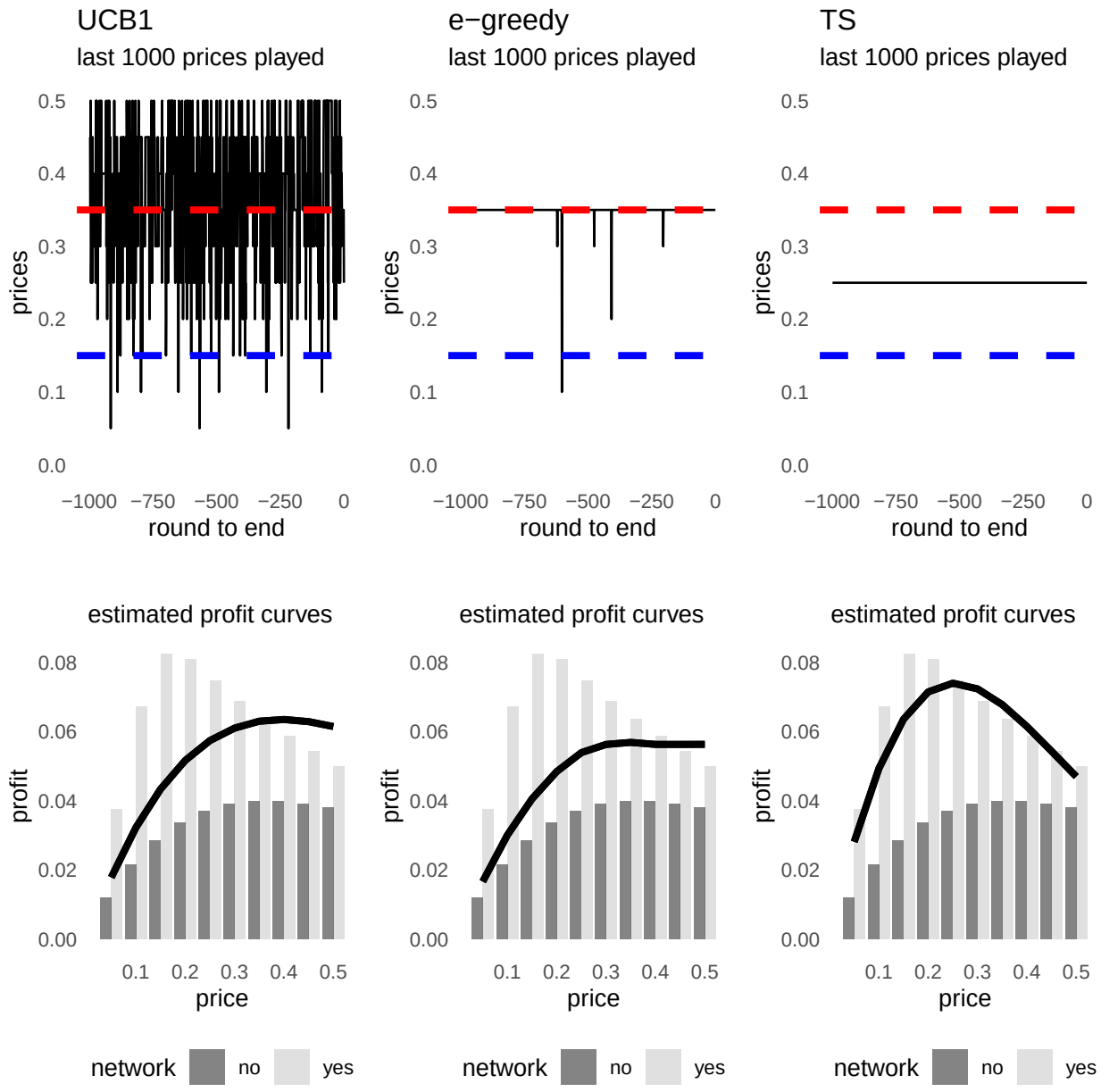

Figure 1: Simulations of mis-specified pricing algorithms under network externalities.

Notes: Simulation outcomes from mis-specified pricing algorithms under direct network externalities. UCB1 (left), ε-greedy algorithms (middle) and Thompson Sampling (right) simulate prices for 1,000,000 rounds and we consider the final 1,000 rounds.

1. Top row of charts shows the final 1,000 prices charged by UCB1 (left), ε-greedy (middle) and Thompson Sampling (right). The blue line shows the optimal price with network effects and the red line shows the optimal price without network effects.

2. Bottom row of charts shows profit functions learned by the UCB1 (left), ε-greedy (middle) and Thompson Sampling (right) algorithm. The light (dark) gray bars show the true profit with (without) network effect. The lines show the estimated profit. For UCB and ε-greedy this is defined as average profit for each price. For Thompson Sampling this is the implied profit curve from the mean demand estimates.

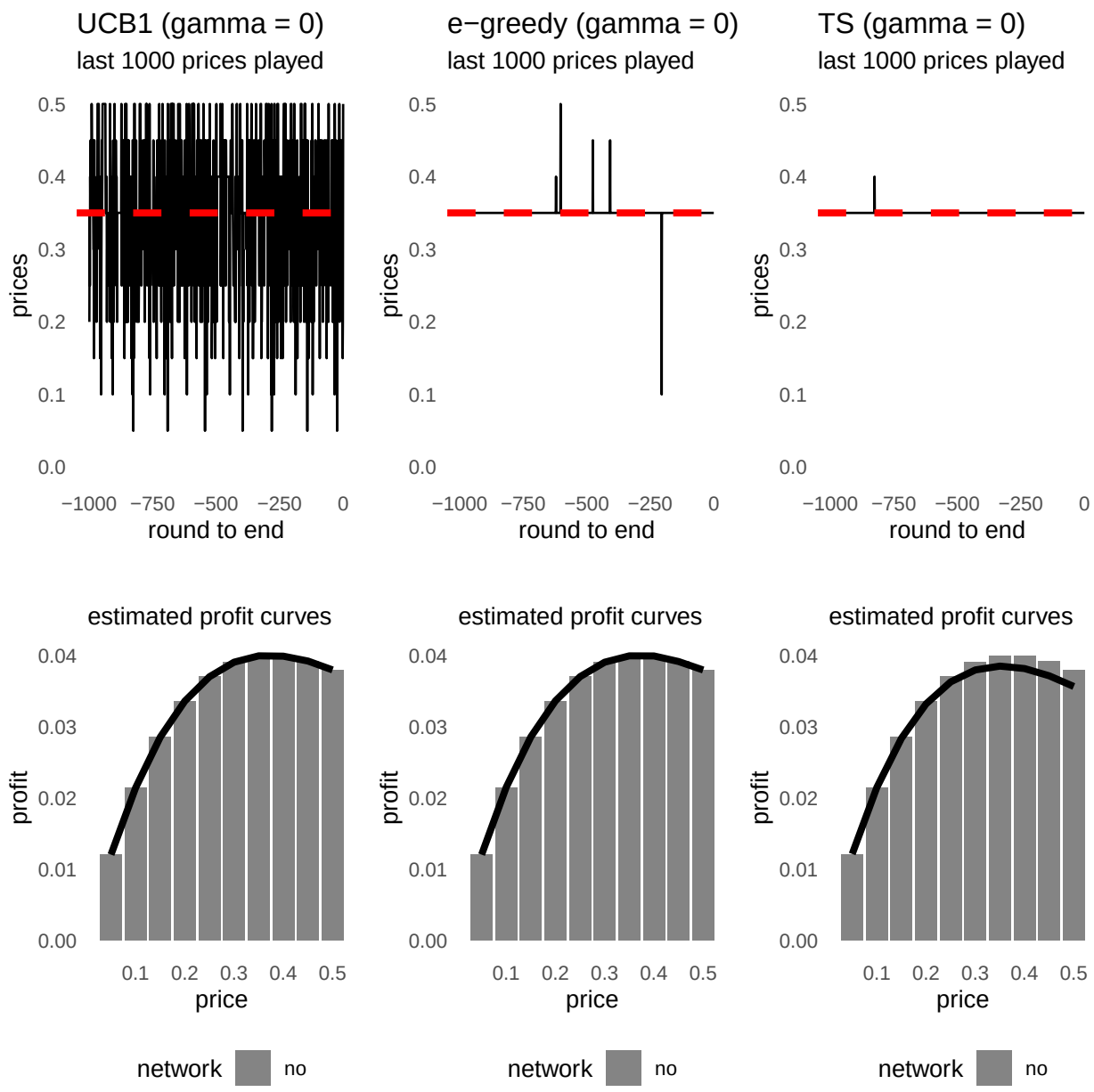

Figure 2: Simulations of correctly specified pricing algorithms.

Notes: Simulation outcomes under correctly specified pricing algorithms without network externalities. UCB1 (left), ε-greedy algorithms (middle) and Thompson Sampling (right) simulate prices for 1,000,000 rounds and we consider the final 1,000 rounds.

1. Top row of charts shows the final 1,000 prices charged by UCB1 (left), ε-greedy (middle) and Thompson Sampling (right). The red line shows the optimal price without network effects.

2. Bottom row of charts shows profit functions learned by the UCB1 (left), ε-greedy (middle) and Thompson Sampling (right) algorithms. The dark gray bars show the true profit without network effects. The lines show the estimated profit. For UCB and ε-greedy this is defined as average profit for each price. For Thompson Sampling this is the implied profit curve from the mean demand estimates.

3.5 Will Platform Pricing Algorithms Reach Collusive Outcomes?

A prominent feature of the algorithmic pricing literature is the potential for pricing algorithms to learn and sustain collusive prices (e.g., Dorner, 2021; Hansen et al., 2021). Collusion is a market failure with serious welfare implications. A natural question arises: if competing platforms adopt platform pricing algorithms, would those algorithms reach collusive outcomes? For clarity, we speak of collusion *between* platforms, not collusion *on* or *within* platforms.

We see two challenges in researching potential algorithmic collusive outcomes between platforms. First, we have argued that the development of platform pricing algorithms faces fundamental challenges and demonstrated that their use may lead to substantial pricing errors if specified poorly. There is no obvious reason to expect that mispricing would disappear when facing a competing platform pricing algorithm. Finding and sustaining fully collusive prices may require both competing platform pricing algorithms to fully resolve the fundamental challenges we have identified. So it seems unlikely that collusive outcomes will occur within the near future.

Second, even defining collusion between platforms is non-trivial. Most insights about coordinated actions among one-sided firms do not translate to multi-sided platforms (Evans and Schmalensee, 2013). For example, Gentzkow et al. (2014) calibrated an empirical model, which then predicted that allowing newspapers to collude in advertising prices would increase total market surplus at the expense of reduced advertiser surplus since newspaper readers generally dislike advertising. Other work predicts that multi-sided collusion is particularly difficult to sustain between multi-sided platforms, due to the interdependence of prices between agent types (Ruhmer, 2010) and non-price attributes (Carbonnel, 2021). Meanwhile, one-sided collusion may lead colluding platforms to compete away the colluding profits on the non-colluding side (Evans and Schmalensee, 2013), and can benefit customers on the collusive side if they generate large enough positive externalities to the non-collusive side (Lefouili and Pinho, 2020).

None of this rules out the possibility of platform pricing algorithms reaching collusive outcomes. The possibility of collusion may be greatest in cases where network externalities play the smallest role, as network externalities are the likely source of pricing algorithm failures within platform settings. However, in cases with substantial or large network externalities, the empirical likelihood of collusion seems to be low in the near future, and will depend on the definition of

platform collusion.

4 Suggestions for Feasible Platform Pricing Algorithms

We have several suggestions for potential solutions to the fundamental challenges of platform pricing algorithms.

First, agents' platform adoption decisions will have to be represented in the state space in some form. We recommend that the platform should seek to measure each agent's value to other agents directly if possible. Depending on the setting, agent value signals might be measurable using combinations of profile views, communications, sales, reviews, or other signals. Then agents could be grouped into manageable numbers of classes based upon interaction value, in order to make the state space small enough to search over, while retaining important dimensions of agent heterogeneity.

Second, when network externalities are large, then the platform knows that retaining valuable agents on the platform increases total platform utility, and repelling value-reducing agents from the platform also increases total platform utility. Both effects work in favor of higher optimal prices. A platform pricing algorithm may find optimal prices faster if it generalizes the objective function to include more than just profits. For example, the platform could specify the pricing objective as a weighted sum of profits and outcomes that agents value directly, such as sales or other interactions.

Relatedly, Johnson et al. (2023) formulate platform utility as a weighted combination of revenue and consumer surplus. They argue the inclusion of consumer surplus directly in the platform's payoff function represents unmodeled platform competition and unmodeled dynamic considerations. Platform uncertainty and desire to learn about unknown customer heterogeneity could offer a third distinct rationale for the platform to maximize value provided alongside platform profit.

Third, a promising approach to simplify the process of learning optimal prices is to consider and model agents' platform adoption and retention decisions. Customers are not endowed with knowledge of platform utility parameters such as ω_i^k, $\beta_{ii'}^k$, and γ_{ij}^k. Platform adoption is a social process through which agents predict platform utility, and then refine those predictions through trial and repeated experience. Studying learning procedural inputs and outputs is likely to help in predicting the resulting purchase behaviors, and likely could speed the algorithmic price learning

process.

Finally, when the state space becomes very large, a straightforward way to limit learning costs is to shrink the action space. A limited number of price points p^b and p^s may help to make state-based learning algorithms more efficient at the expense of exploitation.

5 Discussion

Platform pricing algorithms face fundamental challenges that distinguish them from standard applications of algorithmic pricing. Our review of algorithms implemented by Walmart, Amazon, ZipRecruiter, and AirBnB reveals that current approaches largely disregard network externalities, despite their central importance to platform economics. Our theoretical analysis explains why: network externalities create an exponentially larger state space, complicate the interpretation of pricing experiments, and introduce interdependencies that violate standard experimental validity assumptions.

These challenges help explain the current nonexistent state of platform pricing algorithms. Many platforms maintain relatively stable prices for extended periods, particularly for platform access. When platforms do implement algorithmic pricing, they often focus on narrow use cases where network effects can be safely ignored or treated as background conditions. The empirical evidence suggests this may be rational: our simulations demonstrate that misspecified algorithms that ignore network effects can converge to severely suboptimal prices.

However, platforms that successfully incorporate network externalities into their pricing algorithms may gain substantial advantages. We suggest several promising approaches: directly measuring and clustering agents based on their network value, expanding objective functions beyond pure profit maximization, and studying the social processes of platform adoption. These approaches may help make the fundamental challenges more tractable, though significant technical and empirical work remains to be done.

As platforms continue to grow in economic importance, development of more sophisticated platform pricing algorithms seems possible. Progress will require advances in both technical capabilities and economic understanding of how network effects shape platform dynamics.

Acknowledgements: For helpful comments, we thank Diego Aparicio, Yufeng Huang, Daniel

Sokol and participants at the Cambridge Handbook on Digital Platforms Symposium. Any mistakes are ours alone.

References

Amazon. Amazon.com Issues Statement Regarding Random Price Testing, September 2000. URL https://press.aboutamazon.com/2000/9/amazon-com-issues-statement-regarding-random-price-testing. [Online; accessed 30. Sep. 2024].

Diego Aparicio and Kanishka Misra. Artificial intelligence and pricing. *Artificial intelligence in marketing*, pages 103–124, 2023.

P Auer. Finite-time analysis of the multiarmed bandit problem, 2002.

Yannis Bakos and Hanna Halaburda. Platform competition with multihoming on both sides: Subsidize or not? *Management Science*, 66(12):5599–5607, 2020.

Steven T Berry. Estimating discrete-choice models of product differentiation. *The RAND Journal of Economics*, pages 242–262, 1994.

Rebecca Brown. Amazon Prime: A Timeline from 2005 to 2020, 2023. URL https://web.archive.org/web/20240112213409/https://uk.pattern.com/blog/amazon-prime-timeline-to-present. [Online; accessed 11. Nov. 2024].

Emilio Calvano, Giacomo Calzolari, Vincenzo Denicolo, and Sergio Pastorello. Artificial intelligence, algorithmic pricing, and collusion. *American Economic Review*, 110(10):3267–3297, 2020.

Alexandre Carbonnel. The two sides of platform collusion. *European Competition Journal*, 17(3): 745–760, 2021.

Juan Camilo Castillo. Who benefits from surge pricing? *Available at SSRN 3245533*, 2023.

Sungjin Cho, Gong Lee, John Rust, and Mengkai Yu. Optimal dynamic hotel pricing. In *2018 Meeting Papers*, volume 179, 2018.

Joe Cooprider and Shima Nassiri. Science of price experimentation at Amazon. *Business Economics*, 58(1):34–41, January 2023. doi: 10.1057/s11369-023-00303-9.

Florian E Dorner. Algorithmic collusion: A critical review. *arXiv preprint arXiv:2110.04740*, 2021.

Jean-Pierre Dubé and Sanjog Misra. Personalized pricing and consumer welfare. *Journal of Political Economy*, 131(1):131–189, 2023.

Dean Eckles, Brian Karrer, and Johan Ugander. Design and Analysis of Experiments in Networks: Reducing Bias from Interference. *Journal of Causal Inference*, 5(1), 2017. doi: 10.1515/jci-2015-0021.

Dean Eckles, Brett R. Gordon, and Garrett A. Johnson. Field studies of psychologically targeted ads face threats to internal validity. *Proc. Natl. Acad. Sci. U.S.A.*, 115(23):E5254–E5255, 2018. doi: 10.1073/pnas.1805363115.

David S Evans and Richard Schmalensee. The antitrust analysis of multi-sided platform businesses. Technical report, National Bureau of Economic Research, 2013.

Ravi Ganti, Matyas Sustik, Quoc Tran, and Brian Seaman. Thompson Sampling for Dynamic Pricing. *arXiv*, 2018. doi: 10.48550/arXiv.1802.03050.

Nikhil Garg and Hamid Nazerzadeh. Driver surge pricing. *Management Science*, 68(5):3219–3235, 2022.

Matthew Gentzkow, Jesse M Shapiro, and Michael Sinkinson. Competition and ideological diversity: Historical evidence from us newspapers. *American Economic Review*, 104(10):3073–3114, 2014.

Bita Hajihashemi, Amin Sayedi, and Jeffrey D Shulman. The perils of personalized pricing with network effects. *Marketing Science*, 41(3):477–500, 2022.

Karsten T Hansen, Kanishka Misra, and Mallesh M Pai. Collusive outcomes via pricing algorithms. *Journal of European Competition Law & Practice*, 12(4):334–337, 2021.

Greg Holden. *Starting an Online Business for Dummies, 6th Edition*. Wiley Publishing, Inc., Hoboken, NJ, 2010. ISBN 978-0-470-60210-2.

Yufeng Huang. Pricing frictions and platform remedies: the case of airbnb. *Available at SSRN 3767103*, 2022.

Justin P. Johnson, Andrew Rhodes, and Matthijs Wildenbeest. Platform Design When Sellers Use Pricing Algorithms. *Econometrica*, 91(5):1841–1879, 2023. doi: 10.3982/ECTA19978.

Mingyu Joo, Dinesh K. Gauri, and Kenneth C. Wilbur. Temporal Distance and Price Responsiveness: Empirical Investigation of the Cruise Industry. *Management Science*, 66:5362–5388, 2020. URL `https://pubsonline.informs.org/doi/10.1287/mnsc.2019.3468`.

Bruno Jullien, Alessandro Pavan, and Marc Rysman. Two-sided markets, pricing, and network effects. In *Handbook of industrial organization*, volume 4, pages 485–592. Elsevier, 2021.

Michael Kearns and Satinder Singh. Finite-sample convergence rates for q-learning and indirect algorithms. *Advances in neural information processing systems*, 11, 1998.

Yassine Lefouili and Joana Pinho. Collusion between two-sided platforms. *International Journal of Industrial Organization*, 72:102656, 2020.

Kanishka Misra, Eric M Schwartz, and Jacob Abernethy. Dynamic online pricing with incomplete information using multiarmed bandit experiments. *Marketing Science*, 38(2):226–252, 2019.

Isabel Ruhmer. Platform collusion in two-sided markets. 2010.

J. Manuel Sanchez-Cartas and Evangelos Katsamakas. AI pricing algorithms under platform competition. *Electron. Commer. Res.*, pages 1–28, February 2024. doi: 10.1007/s10660-024-09821-w.

David Smith and Steven Tadelis. Algorithmic Pricing: What Every Antitrust Lawyer Needs to Know. *The Price Point, the newsletter of the American Bar Association Section of Antitrust Law Pricing Conduct Committee*, 22(1), 2021.

Tat-How Teh, Chunchun Liu, Julian Wright, and Junjie Zhou. Multihoming and oligopolistic platform competition. *American Economic Journal: Microeconomics*, 15(4):68–113, 2023.

Hal R Varian. Position auctions. *international Journal of industrial Organization*, 25(6):1163–1178, 2007.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

Kenneth C. Wilbur, Linli Xu, and David Kempe. Correcting Audience Externalities in Television Advertising. *Marketing Science*, 32:827–1011, 2013. URL `https://pubsonline.informs.org/doi/abs/10.1287/mksc.2013.0807`.

Chiwei Yan, Helin Zhu, Nikita Korolko, and Dawn Woodard. Dynamic pricing and matching in ride-hailing platforms. *Naval Research Logistics (NRL)*, 67(8):705–724, 2020.

Peng Ye, Julian Qian, Jieying Chen, Chen-hung Wu, Yitong Zhou, Spencer De Mars, Frank Yang, and Li Zhang. Customized regression model for airbnb dynamic pricing. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 932–940, 2018.

Georgios Zervas, Davide Proserpio, and John W Byers. The rise of the sharing economy: Estimating the impact of airbnb on the hotel industry. *Journal of marketing research*, 54(5):687–705, 2017.

Shunyuan Zhang, Nitin Mehta, Param Vir Singh, and Kannan Srinivasan. Frontiers: Can an artificial intelligence algorithm mitigate racial economic inequality? an analysis in the context of airbnb. *Marketing Science*, 40(5):813–820, 2021.