From Stream to Pool: Dynamic Pricing Beyond i.i.d. Arrivals

Anonymous Author
Anonymous Institution

Abstract

The dynamic pricing problem has been extensively studied under the **stream** model: A stream of customers arrives sequentially, each with an independently and identically distributed valuation. However, this formulation is not entirely reflective of the real world. In many scenarios, high-valuation customers tend to make purchases earlier and leave the market, leading to a *shift* in the valuation distribution. Thus motivated, we consider a model where a **pool** of n non-strategic unit-demand customers interact repeatedly with the seller. Each customer monitors the price intermittently according to an independent Poisson process and makes a purchase if the observed price is lower than her *private* valuation, whereupon she leaves the market permanently. We present a minimax *optimal* algorithm that efficiently computes a non-adaptive policy which guarantees a $1/k$ fraction of the optimal revenue, given any set of k prices. Moreover, we present an adaptive *learn-then-earn* policy based on a novel *debiasing* approach, and prove an $\tilde{O}(kn^{3/4})$ regret bound. We further improve the bound to $\tilde{O}(k^{3/4}n^{3/4})$ using martingale concentration inequalities.

1 Introduction

Pricing with unknown demand is a fundamental challenge in revenue management. Consider the sale of new clothing lines. Each customer visits the (online or offline) store intermittently depending on their availability and makes a purchase if the observed price is lower than her valuation. As each customer typically needs only one unit, she exits the market once a purchase is made. As the product is newly introduced, the seller has little information about customers' valuations to inform their pricing strategy upfront.

Most existing work on dynamic pricing employs what we call a **stream** model: A stream of customers arrives sequentially, each with an independent identically distributed (i.i.d.) valuation. Demand uncertainty is well understood under this model; see, e.g., [Kleinberg and Leighton, 2003, Besbes and Zeevi, 2009] and [Babaioff et al., 2015].

However, the stream model is lacking in many real-world scenarios. In the above clothing example, the demands over time are *neither identical nor independent*. They are not identically distributed since a high-value customer tends to make a purchase early and subsequently leaves the market, resulting in a *shift* in the distribution of valuations towards the lower end.

It should be noted that demand *non-stationarity* has been extensively studied ([Besbes and Zeevi, 2011], [Besbes and Sauré, 2014] and [Den Boer, 2015]). However, the non-stationarity in these work is *exogenous*: It serves to incorporate external factors such as seasonality or promotion and does not depend on the seller's action. In contrast, the non-stationarity in our example is *endogenously* determined by the seller's actions.

Orthogonal to non-stationarity, the *independence* assumption is also questionable. In the stream model, the demand in every time period is independent of the previous prices (even if the demand function is non-stationary over time). However, this is not true in the previous example. To see why the demand rate depends on previous prices, suppose all customers have a valuation 0.5 and *always* monitor the price. Then, the demand rate is 0 at price 1 if and only if the price has *ever* been lower than 0.5.

This problem is also related to *Reinforcement Learning* (RL) for *Partially Observable Markov Decision Process (POMDP)*. In fact, we can encode the state using (i) the remaining time and (ii) the remaining customers in each valuation group. Moreover, we only observe the total number of sales (across all valuation groups) in any interval of time which only gives partial information about the current state. However, known

Preliminary work. Under review by AISTATS 2024. Do not distribute.

results for learning POMDPs are not applicable since they (i) require special structures that do not hold in our problem, (ii) usually rely on revisiting the states, which is infeasible here as the state evolution is unidirectional, and (iii) do not leverage the special structure of our problem. We provide a detailed discussion in the literature review; see Section 1.2.

In order to address these challenges, we consider a single-item revenue maximization problem where a **pool** of unit-demand, non-strategic customers interact repeatedly with a single seller. Each customer monitors the price intermittently according to an independent Poisson process and makes a purchase if she observes a price lower than her private valuation, whereupon she leaves the market permanently. We design an efficient algorithm that computes a nearly optimal non-adaptive policy for the unknown demand. Furthermore, we also propose a learn-then-earn policy with vanishing regret.

1.1 Our Contribution

We initiate the study of dynamic pricing under a *pool-based* model and present the following results.

1. **A Novel Model.** We introduce a novel *pool*-based pricing model: Each customer monitors the price according to an independent Poisson process, makes a purchase when the observed price is below the valuation, and leaves the market *permanently*. In contrast to the *stream*-based model in most existing work, our model better encapsulates the key features of many retailing scenarios where the customers have unit demand. We show that this problem is tractable if the instance is known through the following results.

a) **Price Monotonicity.** We show that the price sequence in any optimal non-adaptive policy is non-increasing; see Proposition 2.3.

b) **Optimal Non-adaptive Policy.** We present an efficient algorithm that computes the optimal non-adaptive policy; see Theorem 2.5.

2. **Optimal Algorithm for Non-adaptive Policy.** We first consider *non-adaptive* policies, i.e., policies that predetermine how the price changes, regardless of observed demands. These policies are particularly compelling and practical because of their operational simplicity. We provide a *complete* settlement of this setting by showing the following results.

a) **A k-Competitive Algorithm.** We present an efficient algorithm that takes a family of instances as input and returns *one* non-adaptive policy. We show that our algorithm is k-competitive for any family of k-price instances, i.e., the output policy is guaranteed to procure a $(1/k)$-fraction of the expected revenue achievable by any (possibly adaptive) policy with *full* knowledge of the true instance; see Theorem 3.1.

b) **A $(1+\log\rho)$-Competitive Algorithm.** The above guarantee is weak for large k. To mitigate this, we propose a variant of our algorithm that restricts its attention to a subset of prices. We show that this algorithm is $(1+\log\rho)$-competitive, where ρ is the ratio between the highest and lowest prices; see Theorem 3.5.

c) **Optimality.** Our algorithm achieves the (maximin) optimal competitive ratio. Specifically, for each $k \geq 1$, we construct a family of k-price instances on which no non-adaptive policy guarantees more than $1/k$ fraction of the optimal revenue on *all* instances in this family; see Theorem 3.6.

3. **Adaptive Policy with Sublinear Regret.** We present an adaptive policy with $\tilde{O}(k^{3/4}n^{3/4})$ regret against the optimal *non-adaptive* policy that knows the size of each valuation group, given any set of k prices. This is achieved by combining the following components.

a) **Learn-then-earn via Debiasing.** We propose a *learn-then-earn* policy that estimates the size of each valuation group. Unlike the stream model (which, in this case, is equivalent to *multi-armed bandits* (MAB)), we face an additional challenge of *confounding* observations: At price p, customers with valuations greater than p may also make a purchase, but we do not observe the valuations of those who made purchases. We devised an unbiased estimator that circumvents this issue by accounting for the (estimated) number of remaining customers from each valuation group. A naive analysis gives an $\tilde{O}(kn^{3/4})$ regret bound; see Theorem 4.2.

b) $o(k)$ **Regret via Martingale Concentration.** Unlike in the stream model, in our problem, a naive analysis only yields *linear* dependence on k. This is essential because the confounding effect *accumulates* over time. As a key technical step, we construct a supermartingale and use the Azuma-Hoeffding inequality to show that the estimation error scales as $\tilde{O}(\sqrt{k})$. This leads to an improved regret bound of $\tilde{O}(k^{3/4}n^{3/4})$.

1.2 Literature Review

Our work is related to the following lines of research.

Dynamic Pricing In the Stream Model. The stream model has been extensively studied since the seminal work by [Gallego and Van Ryzin, 1994] which focused on characterizing the optimal policy with a known demand model. The problem is particularly intriguing when the demand model is unknown, where the seller must balance learning and earning [Kleinberg and Leighton, 2003]. Various fundamental aspects have also been investigated, including finite inventory [Besbes and Zeevi, 2009], [Babaioff et al., 2015], joint inventory-pricing control [Chen and Simchi-Levi, 2004], customer choice model [Broder and Rusmevichientong, 2012], person-

alization [Ban and Keskin, 2021], non-stationarity [Besbes and Zeevi, 2011], just to name a few. For a comprehensive overview, the reader can refer to the survey by [den Boer, 2015]. Although the stream model is broadly applicable in many contexts, in this work we aim to understand the pricing problem from an alternative perspective through the pool-based model.

Pricing with Repeated Interactions. In the stream model, each customer interacts with the seller only once. On the other hand, there is substantial literature where customers engage with the seller multiple times, as in our model. However, these studies differ from ours in two critical ways: (i) they are dedicated primarily to analyzing customers' strategic behavior, often assuming known model dynamics, and (ii) they focus on characterizing the market equilibrium rather than finding policies with provable guarantees. For example, [Besanko and Winston, 1990] considered a pool-based model similar to ours but focused on characterizing subgame perfect Nash equilibrium. [Su, 2007] assumed that the customers are impatient, available from the beginning, and strategically wait for markdowns. [Correa et al., 2016] also considered the pool-based model, but focused on *pre-announced* pricing policies for forward-looking customers. [Wang, 2016] studied the reference effect in intertemporal pricing where customer utility depends on past prices.

Markdown Pricing. As we will soon see, any non-adaptive policy in our problem has a non-increasing price sequence. In revenue management, these policies are often referred to as *price skimming* or *markdown* policies. Existing work usually assumes that the demand model is known. The pool-based model has been extensively studied in the special case of $\lambda = \infty$; see, e.g., Section 5.5.1 of [Talluri and Van Ryzin, 2006]. Furthermore, [Smith and Achabal, 1998, Caro and Gallien, 2012, Heching et al., 2002] considered markdown optimization under known demand. There is also a recent line of research that studies markdown policies with *unknown* demand models see, e.g., [Chen, 2021, Jia et al., 2021] and [Jia et al., 2022]. Unlike our work, these works view monotonicity as a *constraint* rather than as a property of the model's optimal solution.

Partially Observable Reinforcement Learning. Our problem can be reformulated as a *Markovian Decision Process* (MDP). In fact, we can characterize the state by (i) the remaining customers in each valuation group and (ii) remaining time. However, a key challenge is that the seller only observes the total demand, but not the demand from each valuation group.

One may introduce a prior distribution and reformulate this problem as a Partially Observable MDP (POMDP). However, classical hardness results suggest that learning in POMDPs can be (both computationally and statistically) intractable even in simple settings [Krishnamurthy et al., 2016]. Recent results for learning POMDP are not applicable to our problem for multiple reasons. First, they (i) require special structures, such as *block MDPs* [Du et al., 2019, Krishnamurthy et al., 2016] or *decodable MDPs* [Efroni et al., 2022] that do not hold in our problem. Second, they usually rely on revisiting states, which is not feasible in our problem, since state evolution is unidirectional - the number of customers can only decrease, and hence we do not observe the same state twice. Finally, our results exploit the structure of our problem which would be ignored by these works.

2 Model and Preliminaries

We now formally describe our model. Consider a finite continuous time horizon, whose length is normalized to 1. There are n customers with private valuations taken from a known set $\{v_i\}_{i\in[k]}$ where $v_1 \geq \ldots \geq v_k$. There are n_i customers in the i-th valuation group, all having valuation v_i. Customer j monitors the price according to an independent Poisson process $(N_s^j)_{s\in[0,1]}$ with a homogeneous rate $\lambda > 0$. An *instance* $\mathcal{I}$ is specified by a tuple $(\lambda, \{n_i\}_{i\in[k]}, \{v_i\}_{i\in[k]})$.

Policy. A pricing *policy* is a stochastic process $X = (X_t)_{t\in[0,1]}$ taking values on V. A policy is required to be *non-anticipating*, i.e., the price depends only on the "history". Formally, this means that X is adapted to the filtration $(\mathcal{F}_t)$ where $\mathcal{F}_t = \sigma(\{N_s^j : j \in [n], s \in [0,t]\})$.

Customer Behavior. Each customer j makes a purchase when the observed price is less equal than her valuation v_j for the *first* time. To formalize this, we suppress j for now and let $(Y_\ell)_{\ell=1,2,\ldots}$ be i.i.d exponential random variables with mean $1/\lambda$, representing the time lags between the monitor events of this customer. Under this notation, $T_\ell := \sum_{i=1}^{\ell} Y_i$ is the time when the ℓ-th time that the price is monitored by the customer. If the price is ever below the valuation, i.e., if $\{\ell \geq 1 : X_{T_\ell} \leq v\} \neq \emptyset$, then a purchase is made at time T_L where $L := \min\{\ell \geq 1 : X_{T_\ell} \leq v\}$. The customer immediately leaves the market once a purchase is made. We now can formally define the revenue.

Definition 2.1 (Revenue)**.** Let $X = (X_s)_{s\in[0,1]}$ be a policy. For each customer $j \in [n]$, let $\tau_j \in [0,1]$ be the time when customer j makes a purchase and set $\tau_j = \infty$ if she never purchases. Then, the (random) *revenue* is $R_X := \sum_{j\in[n]} X_{\tau_j} \cdot \mathbf{1}(\tau_j \leq 1)$.

A compelling class of policies is the class of non-adaptive policies, where the prices are determined upfront, regardless of the purchase events. These policies

are widely applied in practice due to their simplicity and effectiveness; see, e.g., [Ma et al., 2021].

Definition 2.2 (Non-adaptive Policy)**.** A policy $(X_s)_{s\in[0,1]}$ is *non-adaptive* if for any s, the random variable X_s is a constant.

2.1 Optimization Under Known Demand

When the sizes of each valuation group are known, the problem is relatively easy to handle, at least in the non-adaptive setting. We first show that the price sequence in any optimal non-adaptive policy is non-increasing over time. A policy with this property is often referred to as a *markdown* policy in revenue management.

Proposition 2.3 (Price Monotonicity)**.** *Suppose $(X_s)_{s\in[0,1]}$ is an optimal non-adaptive policy. Then, $X_s \geq X_t$ almost surely (a.s.) whenever $0 \leq s < t \leq 1$.*

This structural result follows from a simple swapping argument. Suppose the price sequence is not non-increasing, say, the price is p_L in some interval $[t-\varepsilon, t]$ and increases to p_H in $[t, t+\varepsilon]$ where $\varepsilon > 0$. We show that the expected revenue does not decrease if we swap prices p_H, p_L in two intervals. To see this, note that customers with valuations lower than p_H are not affected by this swap, since they can only buy the product at price p_L in the time interval $[t-\epsilon, t+\epsilon]$. On the other hand, we can argue that if a customer has a valuation higher than p_H, then after the swap she is more likely to purchase at price p_H.

We will therefore restrict our attention to non-adaptive markdown policies subsequently. Each policy in this class can be specified by a sequence $(t_1, \cdots, t_k)$ where the policy selects the price v_i from time t_i to t_{i+1}. Conveniently, we have a closed-form formula for the expected revenue for any non-adaptive markdown policy.

Proposition 2.4 (Expected Revenue of Markdown Policy)**.** *For any instance $\mathcal{I} = (\lambda, \{n_i\}_{i=1}^k, \{v_i\}_{i=1}^k)$ and non-adaptive markdown policy $\pi = (t_i)_{i\in[k]}$, define the revenue function $\mathrm{Rev}(\pi, \mathcal{I})$ as*

$$\sum_{i\in[k]} n_i \sum_{j:i\leq j\leq k} v_j e^{-\lambda(t_j - t_i)} \left(1 - e^{-\lambda(t_{j+1} - t_j)}\right).$$

where $t_{k+1} := 1$. Then,

$$\mathbb{E}[R_\pi] = \mathrm{Rev}(\pi, \mathcal{I}).$$

Each term in the inner summation corresponds to the expected revenue from a customer with valuation v_i in the j-th time interval. The term $e^{-\lambda(t_j - t_i)}$ is the probability that a customer of valuation v_i remains in the market until time t_j, and $1 - e^{-\lambda(t_j - t_i)}$ is the probability that the customer makes a purchase during the j-th interval, assuming that she is still in the market. Monotonicity enables us to compute an optimal non-adaptive policy.

Theorem 2.5 (Optimal Non-adaptive Policy)**.** *There is a polynomial time algorithm that computes an optimal non-adaptive policy for any instance $\mathcal{I} = (\lambda, \{n_i\}_{i=1}^k, \{v_i\}_{i=1}^k)$.*

So far, we have shown that our problem is tractable if the instance is known. In Sections 3 and 4 we consider the scenario where the instance is unknown.

3 Competitive Non-adaptive Policy

For new products, the seller usually only has incomplete knowledge about the true model. An important class of policies is non-adaptive policies, i.e., policies that predetermine how price trajectory regardless of realized purchases. Non-adaptive policies are widely applied in the real world due to their operational simplicity; see Section 5 of [Talluri and Van Ryzin, 2006].

In this section, we consider how to compute a *non-adaptive* policy given only the monitoring rate and the price space. We provide a *complete* settlement of this setting by presenting an algorithm that computes a non-adaptive policy that guarantees a best-possible $1/k$ fraction of the optimal revenue. For any instance $\mathcal{I}$, denote by $\mathrm{OPT}(\mathcal{I})$ the optimal revenue achievable by any non-adaptive policy.

Theorem 3.1 (Competitive Ratio Lower Bound)**.** *There is an algorithm that takes as input the price space $\{v_i\}_{i\in[k]}$, the monitoring rate λ, and computes in polynomial time a non-adaptive policy π such that for any instance $\mathcal{I} = (\lambda, \{n_i\}_{i=1}^k, \{v_i\}_{i=1}^k)$, we have*

$$\frac{\mathrm{Rev}(\pi, \mathcal{I})}{\mathrm{OPT}(\mathcal{I})} \geq \frac{1}{k}.$$

We outline the proof and defer the details to the appendix. A natural idea is to write $\mathrm{Rev}(\pi, \mathcal{I})/\mathrm{OPT}(\mathcal{I})$ as a function $f(t_1, \ldots, t_k; n_1, \ldots, n_k)$ and then solve a *bilevel* program

$$\text{(BP1)} \max_{t_1,\ldots,t_k} \min_{n_1,\ldots,n_k} f(t_1, \ldots, t_k; n_1, \ldots, n_k),$$
$$\text{such that} \quad 0 \leq t_i \leq t_j \leq 1, \forall i < j,\ i, j \in [k].$$

However, this approach fails since most results on bilevel optimization assume certain structures such as concavity-convexity, but our f is neither convex in the n_i's nor concave in the t_i's.

3.1 Upper Bounding on the Optimal Revenue

An alternative idea is to find a closed-form formula for the denominator for any given (n_i)'s, and reduce

the bilevel problem to a single-level problem. However, this approach does not work either since finding a closed-form solution for $\mathrm{OPT}(\mathcal{I})$ is a formidable task. To circumvent this, we introduce the following upper bound on $\mathrm{OPT}(\mathcal{I})$.

Lemma 3.2 (Upper Bound on $\mathrm{OPT}(\mathcal{I})$)**.** *For any instance $\mathcal{I} = (\lambda, \{n_i\}_{i=1}^k, \{v_i\}_{i=1}^k)$, we define $\mathrm{UB}(\mathcal{I}) := \sum_{i\in[k]} n_i v_i \cdot (1 - e^{-\lambda})$. Then, for any policy π, we have*

$$\mathbb{E}[R_\pi] \le \mathrm{UB}(\mathcal{I}).$$

To see this, note that if a customer has valuation v, then the maximum expected revenue from this customer is at most $v(1 - e^{-\lambda})$, which is attained by the policy that always selects price v. The expression $\mathrm{UB}(\mathcal{I})$ is simply the sum of this upper bound over all customers.

On the other hand, it should be noted that $\mathrm{UB}(\mathcal{I})$ can be much greater than $\mathrm{OPT}(\mathcal{I})$. In fact, the $\mathrm{UB}(\mathcal{I})$ is attained by a *personalized* policy, i.e., prices for different customers may differ, whereas $\mathrm{OPT}(\mathcal{I})$ is defined over *non-personalized* policies.

3.2 Linearization

With this upper bound, we next focus on the bilevel optimization problem where $\mathrm{OPT}(\mathcal{I})$ is replaced with $\mathrm{UB}(\mathcal{I})$. Explicitly, we consider

$$\begin{aligned}(\mathrm{BP2}) \quad &\max_{t_1,\dots,t_k} \min_{n_1,\dots,n_k} \frac{\mathrm{Rev}(\pi,\mathcal{I})}{\mathrm{UB}(\mathcal{I})},\\ \text{such that} \quad &0 \le t_i \le t_j \le 1, \forall i < j,\ i,j \in [k].\end{aligned}$$

The above bilevel problem is still not readily solvable since $\mathrm{Rev}(\pi,\mathcal{I})$ and $\mathrm{UB}(\mathcal{I})$ are both *non-linear* functions. To circumvent this, we consider a *linear surrogate* for each of them, motivated by Taylor's expansion.

Definition 3.3 (Linear Surrogate)**.** For any instance $\mathcal{I} = (\lambda, \{n_i\}_{i\in[k]}, \{v_i\}_{i\in[k]})$ and non-adaptive policy $\pi = (t_i)$, we define the linear surrogate of $\mathrm{UB}(\mathcal{I})$ and $\mathrm{Rev}(\pi,\mathcal{I})$ as

$$\begin{aligned}\mathrm{UB}'(\mathcal{I}) &:= \sum_{i\in[k]} n_i v_i \lambda,\\ \mathrm{Rev}'(\pi,\mathcal{I}) &:= \sum_{i\in[k]} n_i \sum_{j\in[k]:j\ge i} \lambda v_j \left(t_{j+1} - t_j\right).\end{aligned}$$

We show that this linearization only decreases the objective in (BP2). Thus, a lower bound on the linearized bilevel program implies a lower bound on (BP2).

Lemma 3.4 (Linearization Reduces the Objective)**.** *For any instance $\mathcal{I} = (\lambda, \{n_i\}_{i=1}^k, \{v_i\}_{i=1}^k)$ and non-adaptive policy π, we have*

$$\frac{\mathrm{Rev}(\pi,\mathcal{I})}{\mathrm{UB}(\mathcal{I})} \ge \frac{\mathrm{Rev}'(\pi,\mathcal{I})}{\mathrm{UB}'(\mathcal{I})}.$$

To see why this is true, observe that the function $h(x) := \frac{1-e^{-x}}{x}$ is decreasing in x. For any positive $x \le y$, we have $(1 - e^{-x})/x \ge (1 - e^{-y})/y$, which rearranges to

$$\frac{1 - e^{-x}}{1 - e^{-y}} \ge \frac{x}{y}.$$

3.3 Reducing to a Linear Program

With Lemma 3.4, now we further simplify the bilevel program (BP2) by replacing the objective with the ratio between the linearized functions. This results in the following bilevel program:

$$\begin{aligned}(\mathrm{BP3}) \quad &\max_{t_1,\dots,t_k} \min_{n_1,\dots,n_k} \frac{\sum_{i\in[k]} n_i \sum_{j\in[k]:j\ge i} v_j \left(t_{j+1} - t_j\right)}{\sum_i n_i v_i},\\ \text{such that} \quad &0 \le t_i \le t_j \le 1, \forall i < j,\ i,j \in [k].\end{aligned}$$

We construct an optimal solution to the (BP3) by reduction to linear program (LP). Observe that the inner minimum is always achieved by a binary vector with exactly one non-zero entry. More precisely, it is given by $n_i = n \cdot \mathbf{1}(i = i^*)$ where

$$i^* = \arg\min \left\{ \frac{\sum_{j=i}^k v_j t_j}{v_i} : i \in [k] \right\}.$$

(For simplicity, we assume i^* is unique; apparently, this is not essential to the analysis.) Thus, (BP3) can be reformulated as

$$\begin{aligned}&\max_{(t_i),c} \quad c\\ \text{such that} \quad &c \le \frac{\sum_{j\in[k]:j\ge i} v_j \left(t_{j+1} - t_j\right)}{v_i}, \forall i \in [k],\\ &0 \le t_i \le t_{i+1} \le 1, \forall i \in [k].\end{aligned}$$

We can easily verify that the optimal solution is attained when all the inequalities are binding. In this case, the optimal solution (t_i^*) satisfies

$$t_{i+1}^* - t_i^* = \left(1 - \frac{v_{i+1}}{v_i}\right)(1 - t_k^*), \quad \forall i < k. \tag{1}$$

This solves to

$$t_k^* = 1 - \frac{1}{k - \sum_{1\le i\le k-1} \frac{v_{i+1}}{v_i}}.$$

Finding t_i^* for $i < k$ can be done with backward substitution using equation (1).

The resulting performance guarantee is given by

$$\mathrm{CR}(v_1,\dots,v_k) = 1 - t_k^* = \frac{1}{k - \sum_{i=1}^{k-1} v_{i+1}/v_i}.$$

So far, we have a performance guarantee for fixed $v_1,\dots,v_k$. Next, we characterize the worst-case performance guarantee overall v_is, i.e., the worst-case competitive ratio. By simple calculation, one can verify that $\mathrm{CR}(v_1,\dots,v_k)$ is at least $1/k$ for any $v_1,\dots,v_k$.

3.4 Competitive Ratio for Small Aspect Ratio

Note that when k grows, the above result gets weaker and weaker. This motivates us to employ a core set for the valuation levels. More precisely, let $a > 0$ and b be the minimum and maximum of all the v_i respectively. For any $\epsilon > 0$, we can partition the interval $[a, b]$ into subintervals $[a(1+\varepsilon)^{j-1}, a(1+\varepsilon)^j)$ for $j = 1$ to $\log(b/a)/\log(1+\varepsilon)$. Further, we compute the non-adaptive policy based on the valuation set $\{a(1+\varepsilon)^{j-1}\}$ for $j = 1$ to $\log(b/a)/\log(1+\varepsilon)$, and derive another competitive ratio bound using these.

Theorem 3.5 (Competitive Ratio Lower Bound)**.** *For any instance $\mathcal{I} = (\lambda, \{n_i\}_{i=1}^k, \{v_i\}_{i=1}^k)$ where $\{n_i\}_{i=1}^k$ is unknown to the seller, we can compute in polynomial time a nonadaptive policy $\pi = (t_1, \cdots, t_{k+1})$ such that*

$$\frac{\mathrm{Rev}(\pi, \mathcal{I})}{\mathrm{OPT}(\mathcal{I})} \geq \frac{1}{1 + \log(v_1/v_k)}.$$

Since the optimal revenue on the whole valuation set, $\mathrm{OPT}(\mathcal{I})$, is at most $(1+\epsilon)$ of the optimal revenue on the core set, the competitive ratio we derive on the core set is at least,

$$\frac{1}{(1+\epsilon)(k - \sum_{i=1}^{k-1} v_{i+1}/v_i)},$$

where $k = \log(v_1/v_k)/\log(1+\varepsilon)$ for the core set, and $v_{i+1}/v_i = 1/(1+\epsilon)$ for $i \in [k-1]$. Plugging the expression of k and v_{i+1}/v_i, then the competitive ratio on the core is

$$\frac{1}{1 + \epsilon \log(v_1/v_k)/\log(1+\varepsilon)}.$$

Note $\epsilon/\log(1+\varepsilon)$ is increasing in ϵ and $\epsilon/\log(1+\varepsilon) = 1$ when ϵ goes to 0, therefore, the competitive ratio is at least,

$$\frac{1}{1 + \epsilon \log(v_1/v_k)/\log(1+\varepsilon)} \geq \frac{1}{1 + \log(v_1/v_k)}.$$

3.5 Upper-Bounding the Competitive Ratio

We also show that the above lower bound of $1/k$ is the best possible. No algorithm can achieve a fraction larger than $1/k$ of the optimal revenue. In Theorem 3.6, we demonstrate that for any non-adaptive policy π, there exists an instance such that the policy can achieve at most $\frac{1}{k} + \epsilon$ fraction of the optimal revenue.

Theorem 3.6 (Upper Bound on Competitive Ratio)**.** *For any integer $k > 0$, $\varepsilon > 0$ and non-adaptive policy π, there exists an instance $\mathcal{I} = \mathcal{I}_{\varepsilon,k}$ such that*

$$\frac{\mathrm{Rev}(\pi, \mathcal{I})}{\mathrm{OPT}(\mathcal{I})} \leq \frac{1}{k} + \epsilon.$$

For small λ, the error from linearization is negligible. For any integer $k > 0$, $\varepsilon > 0$ and non-adaptive policy π, we only need to construct $\{v_i\}_{i=1}^k$ such that the competitive ratio, $\mathrm{CR}(v_1, \ldots, v_k) = 1/(k - \sum_{i=1}^{k-1} v_{i+1}/v_i)$ goes to $1/k$. Consider a geometric sequence with $v_1 = 1$, $v_{i+1} = \phi v_i$ for $i \in [k-1]$. For any $\varepsilon > 0$, there exist a ϕ such that the ratio $1/(k - \sum_{i=1}^{k-1} v_{i+1}/v_i) \leq \frac{1}{k} + \epsilon$, i.e., the ratio $1/k$ is tight.

4 Low-Regret Adaptive Policy

Now we consider adaptive policies in the presence of unknown demand. Specifically, we only assume knowledge of the total number of customers n, the price levels, and the monitoring rate λ, but not the number of customers n_i in each valuation group. Our policies aim to simultaneously learn the demand and optimize pricing decisions given a finite time horizon. As is standard in the demand learning literature, we will analyze the *regret* of our policy. We consider the optimal non-adaptive policy as our benchmark and denote its expected revenue under instance $\mathcal{I}$ as $\mathrm{OPT}(\mathcal{I})$. Thus we define the worst-case regret as follows.

Definition 4.1 (Worst-case Regret)**.** For a policy π, we define its worst-case regret is

$$\mathrm{Regret}(\pi) := \sup_{\mathcal{I}} \{\mathrm{OPT}(\mathcal{I}) - \mathrm{Rev}(\pi, \mathcal{I})\}$$

where the supremum is taken over all instances $\mathcal{I}$ where the total number of customers n, price levels $\{v_i\}_{i=1}^k$, and monitoring rate λ are fixed. The demand at each price level n_i may vary arbitrarily subject to the constraint $\sum_{i \in [k]} n_i = n$.

As the main result of this section, we present an adaptive policy with sublinear regret in both n and k.

Theorem 4.2 (Adaptive Policy with Sublinear Regret)**.** *There exists an adaptive policy π^{LTE} which does not know the demand at each price level which satisfies* $\mathrm{Regret}(\pi^{\mathrm{LTE}}) = \tilde{O}(k^{3/4} \cdot n^{3/4})$.

Our regret bound is higher than the optimal $\tilde{\Theta}(\sqrt{kn})$ regret bound [Auer et al., 2002] for the stream model (which is equivalent to k-armed bandits). This is because in the stream model, the effect of exploration is *local* in the sense that what the seller does in an interval of time only affects the customers arriving in that interval. In contrast, the effect of an action in our model is global: Regardless of how long an action lasts, it can potentially affect $\Omega(n)$ customers. Therefore, it is reasonable to not expect the same order of regret as in the stream model.

Our policy is formally described below in Algorithm 1. The policy operates in two phases. Initially, we explore each of the price levels $v_1, v_2, \ldots, v_{k-1}$ for fixed

intervals of length $s_1, s_2, \ldots, s_{k-1}$. When exploring the i-th price level, we keep track of the realized demand D_i, which we use to construct estimates $\{\hat{n}_i\}_{i\in[k]}$ of the original demand. From there we construct an estimated instance $\hat{\mathcal{I}} = (\lambda, \{\hat{n}_i\}_{i\in[k]}, \{v_i\}_{i\in[k]})$, and compute an optimal non-adaptive policy for this instance only a shortened horizon of length $1 - s_{\text{sum}}$ where $s_{\text{sum}} = \sum_{i=1}^{k-1} s_i$ is the total exploration time.

Algorithm 1: Learn-then-Earn Policy

Data: Partial Instance $(n, \{v_i\}_{i=1}^k, \lambda)$, Exploration times $(s_1, s_2, \ldots, s_{k-1})$
Result: Policy π^{LTE}

`//Learning phase`
for $i = 1, 2, \ldots, k-1$ **do**
  Use price v_i for time s_i
  Observe sales D_i
end
`//Construct estimates` $\{\hat{n}_i\}_{i=1}^k$
Define the function $q(x) = 1 - \exp(-\lambda x)$
for $i = 1, 2, \ldots, k-1$ **do**
  $\hat{n}_i \leftarrow \frac{D_i}{q(s_i)} - \sum_{j<i}(\hat{n}_j - D_j)$
end
$\hat{n}_k \leftarrow n - \sum_{i<k} \hat{n}_i$
`//Earning Phase`
$\hat{\mathcal{I}} \leftarrow (\lambda, \{\hat{n}_i\}_{i\in[k]}, \{v_i\}_{i\in[k]})$
$s_{\text{sum}} \leftarrow \sum_{i=1}^{k-1} s_i$
Find optimal non-adaptive policy $\hat{t}_1, \ldots, \hat{t}_k$ for $\hat{\mathcal{I}}$ on the time interval $[0, 1 - s_{\text{sum}}]$[1]
for $i = 1, 2, \ldots, k$ **do**
  Use price v_i during times $[\hat{t}_i + s_{\text{tot}}, \hat{t}_{i+1} + s_{\text{tot}}]$
end

4.1 Debiasing the Demand

As is standard in MAB, we aim to construct *unbiased* estimates of the model parameter. In Algorithm 1, we use price v_1 for time s_1 and track (random) demand D_1 during this period. Recall that each customer monitors the price with Poisson rate λ, so $D_1 \sim \text{Binomial}(n_1, q(s_1))$ where $q(x) := 1 - \exp(-\lambda x)$. Thus, $D_1/q(s_1)$ is an unbiased estimate of n_1.

However, when we explore prices $v_2, v_3, \ldots, v_{k-1}$, the situation is more complicated. There may still be active customers (i.e., customers who have not exited the market) with a valuation v_1 that purchase during this time, *confounding* the observed demand $D_2, D_3, \ldots$ in future stages.

As the pivotal step, we develop a novel unbiased estimator that overcomes this issue. Starting with $\hat{n}_1 = n_1$, for each $i = 2, 3, \ldots, k$, we recursively define

$$\hat{n}_i = \frac{D_i}{q(s_i)} - \sum_{j<i}(\hat{n}_j - D_j), \tag{2}$$

The first part is similar to the naive estimator we used for $\hat{n}_1$, while the second part aims to remove the confounding affect of customers at higher valuations. We show that this estimator is unbiased.

Lemma 4.3 (Unbiasedness)**.** *Let $s_1, s_2, \ldots, s_{k-1}$ be the lengths of each exploration period and $D_1, D_2, \ldots, D_{k-1}$ be the realized demands. For $i = 1, \ldots, k$, we have $\mathbb{E}[\hat{n}_i] = n_i$.*

As a quick sketch, we show this by induction on $i < k$. The base case is obvious. For the inductive case $1 < i < k$, we observe that conditioned on D_j for $j < i$, we have $D_i \sim \text{Bin}(\sum_{j\le i} n_j - \sum_{j<i} D_j, q(s_i))$. Using this we can show $\mathbb{E}[\hat{n}_i] = n_i + \sum_{j<i} \mathbb{E}[\hat{n}_j - n_j]$, which equals n_i under the inductive hypothesis.

4.2 The Case of Two Price Levels

We demonstrate the main ideas by showing a regret bound of $\tilde{O}(n^{3/4})$ in the two-price case. We specify an exploration time $s \in [0, 1]$, then set the price $X_t = v_1$ for all $t \le s$. Let D be the demand (number of sales) observed in this period. As discussed in Section 4.1, we use $\hat{n}_1 = D/q(s)$ and $\hat{n}_2 = n - \hat{n}_1$ as unbiased estimates of n_1 and n_2. Using these, we construct the estimated instance $\hat{\mathcal{I}} = (\lambda, \{\hat{n}_1, \hat{n}_2\}, \{v_1, v_2\})$ and compute a non-adaptive policy $\hat{\pi}$ achieving revenue $\text{OPT}(\hat{\mathcal{I}})$ for the remaining time horizon $1 - s$ and follow it.

We decompose regret into two quantities that we bound separately. In addition to $\hat{\mathcal{I}}$ as defined above, define $\mathcal{I}' = (\lambda, \{n_1 - D, n_2\}, \{v_1, v_2\})$ as the instance which remains after observing the demand D. We decompose the regret as follows.

Lemma 4.4 (Regret Decomposition)**.** *Define $\eta_1 = |\mathbb{E}[\text{Rev}(\hat{\pi}, \mathcal{I}')] - \mathbb{E}[\text{OPT}(\hat{\mathcal{I}})]|$ and $\eta_2 = |\mathbb{E}[\text{OPT}(\hat{\mathcal{I}})] - \text{OPT}(\mathcal{I})|$. Then,*

$$\text{Regret}(\pi^{\text{LTE}}) \le \eta_1 + \eta_2.$$

The proof follows straightforwardly by noting that $\mathbb{E}[\text{Rev}(\hat{\pi}, \mathcal{I}')]$ is a lower bound on the revenue of our policy since it only accounts for the revenue in the *earning* phase.

We will show that for suitable s, both terms above are $\tilde{O}(n^{3/4})$. To this end, we first show that η_1 will grow linearly in s. For this we use two observations. First, observe that our estimates are unbiased and the revenue is linear in the size of each valuation group. Second, we observe that the impact of the exploration phase (which has length s) is linear in s.

[1] This can be done using the dynamic programming algorithm from Theorem 2.5 by rescaling time to this interval.

Lemma 4.5 (Analysis of η_1)**.** *We have* $\eta_1 = O(\lambda n v_1 s)$.

To bound η_2, we use concentration inequalities to show that our estimates are close to the target values with high probability.

Lemma 4.6 (Analysis of η_2)**.** *For our policy* π^{LTE}*, we have* $\eta_2 = O(v_1\sqrt{n\log(n)}/\lambda s) + o(1)$.

At a high level, we apply Hoeffding's inequality to $D \sim \mathrm{Binomial}(n_1, q(s))$, and combine this with the approximation $q(s) = 1 - \exp(-\lambda s) \approx \lambda s$ for small λs.

The following lemma states that if two functions are point-wise close, then their maximums are also close. This essentially follows from the triangle inequality.

Lemma 4.7. *Let* f, g *be real-valued functions defined on any set* $\mathcal{X}$*. If for all* $x \in \mathcal{X}$*, we have* $|f(x) - g(x)| \le \epsilon$*, then* $|\max_x f(x) - \max_x g(x)| \le 3\epsilon$.

Lemma 4.6 then follows by choosing the functions $f = \mathrm{Rev}(\cdot, \hat{\mathcal{I}})$ and $g = \mathrm{Rev}(\cdot, \mathcal{I})$, and choose ε to be the bound implied by Hoeffding's inequality.

Now we complete the analysis for the two-price case. From Lemma 4.4, Lemma 4.5 and Lemma 4.6 we have

$$\mathrm{Regret}(\pi^{\mathrm{LTE}}) \le O\left(\lambda n v_1 s + \frac{v_1\sqrt{n\log(n)}}{\lambda s}\right) + o(1).$$

The $\tilde{O}(n^{3/4})$ bound follows by taking $s = \tilde{\Theta}(n^{-1/4}/\lambda)$.

4.3 Extending to k Price Levels

We briefly sketch how we extend the analysis from two price levels to k price levels.

Our current analysis for the two-price setting only leads to a bound that depends linearly on k. To achieve a sublinear dependence on k, we need to be more careful in our analysis of the *total* error in our estimates $\hat{n}_i$. Due to the dependencies that exist between our estimates $\hat{n}_i$, we cannot directly apply concentration inequalities to control the total error. Instead, we employ a more careful analysis, showing that the sequence $Z_i = \sum_{j\le i}(\hat{n}_j - n_j) - \alpha_i$ is a supermartingale for an appropriate choice of $\alpha_i > 0$. Then, we apply the Azuma-Hoeffding inequality to obtain a bound that is sublinear in k for the total error. Using this in the rest of our analysis leads to the $\tilde{O}(k^{3/4}n^{3/4})$ bound on the regret. We defer the details to the appendix.

5 Future Work

This work opens up a wealth of new directions and open problems.
1. Lower bounds for the adaptive setting: Known techniques for deriving regret lower bounds for MAB turn out to be ineffective for our problem, and we have to develop new proof strategies.
2. Unknown λ: In reality, the monitoring rate λ may be unknown and must also be learned online. It is not clear how to generalize our LTE policy to handle unknown λ.
3. Inventory constraint: The problem becomes substantially harder if the inventory is finite, which caps our learning process.
4. New arrivals: In reality, there may be new arrivals apart from the initial group of customers, making the problem significantly harder. For example, in this case the monotonicity result no longer holds.

References

[Auer et al., 2002] Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256.

[Babaioff et al., 2015] Babaioff, M., Dughmi, S., Kleinberg, R., and Slivkins, A. (2015). Dynamic pricing with limited supply.

[Ban and Keskin, 2021] Ban, G.-Y. and Keskin, N. B. (2021). Personalized dynamic pricing with machine learning: High-dimensional features and heterogeneous elasticity. *Management Science*, 67(9):5549–5568.

[Besanko and Winston, 1990] Besanko, D. and Winston, W. L. (1990). Optimal price skimming by a monopolist facing rational consumers. *Management science*, 36(5):555–567.

[Besbes and Sauré, 2014] Besbes, O. and Sauré, D. (2014). Dynamic pricing strategies in the presence of demand shifts. *Manufacturing & Service Operations Management*, 16(4):513–528.

[Besbes and Zeevi, 2009] Besbes, O. and Zeevi, A. (2009). Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57(6):1407–1420.

[Besbes and Zeevi, 2011] Besbes, O. and Zeevi, A. (2011). On the minimax complexity of pricing in a changing environment. *Operations research*, 59(1):66–79.

[Broder and Rusmevichientong, 2012] Broder, J. and Rusmevichientong, P. (2012). Dynamic pricing under a general parametric choice model. *Operations Research*, 60(4):965–980.

[Caro and Gallien, 2012] Caro, F. and Gallien, J. (2012). Clearance pricing optimization for a fast-fashion retailer. *Operations research*, 60(6):1404–1422.

[Chen, 2021] Chen, N. (2021). Multi-armed bandit requiring monotone arm sequences. *Advances in Neural Information Processing Systems*, 34:16093–16103.

[Chen and Simchi-Levi, 2004] Chen, X. and Simchi-Levi, D. (2004). Coordinating inventory control and pricing strategies with random demand and fixed ordering cost: The finite horizon case. *Operations research*, 52(6):887–896.

[Correa et al., 2016] Correa, J., Montoya, R., and Thraves, C. (2016). Contingent preannounced pricing policies with strategic consumers. *Operations Research*, 64(1):251–272.

[den Boer, 2015] den Boer, A. V. (2015). Dynamic pricing and learning: historical origins, current research, and new directions. *Surveys in operations research and management science*, 20(1):1–18.

[Den Boer, 2015] Den Boer, A. V. (2015). Tracking the market: Dynamic pricing and learning in a changing environment. *European journal of operational research*, 247(3):914–927.

[Du et al., 2019] Du, S., Krishnamurthy, A., Jiang, N., Agarwal, A., Dudik, M., and Langford, J. (2019). Provably efficient rl with rich observations via latent state decoding. In *International Conference on Machine Learning*, pages 1665–1674. PMLR.

[Efroni et al., 2022] Efroni, Y., Jin, C., Krishnamurthy, A., and Miryoosefi, S. (2022). Provable reinforcement learning with a short-term memory. In *International Conference on Machine Learning*, pages 5832–5850. PMLR.

[Gallego and Van Ryzin, 1994] Gallego, G. and Van Ryzin, G. (1994). Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management science*, 40(8):999–1020.

[Heching et al., 2002] Heching, A., Gallego, G., and van Ryzin, G. (2002). Mark-down pricing: An empirical analysis of policies and revenue potential at one apparel retailer. *Journal of revenue and pricing management*, 1(2):139–160.

[Jia et al., 2021] Jia, S., Li, A., and Ravi, R. (2021). Markdown pricing under unknown demand. *Available at SSRN 3861379*.

[Jia et al., 2022] Jia, S., Li, A. A., and Ravi, R. (2022). Dynamic pricing with monotonicity constraint under unknown parametric demand model. In *Advances in Neural Information Processing Systems*.

[Kleinberg and Leighton, 2003] Kleinberg, R. and Leighton, T. (2003). The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 594–605. IEEE.

[Krishnamurthy et al., 2016] Krishnamurthy, A., Agarwal, A., and Langford, J. (2016). Pac reinforcement learning with rich observations. *Advances in Neural Information Processing Systems*, 29.

[Ma et al., 2021] Ma, W., Simchi-Levi, D., and Zhao, J. (2021). Dynamic pricing (and assortment) under a static calendar. *Management Science*, 67(4):2292–2313.

[Smith and Achabal, 1998] Smith, S. A. and Achabal, D. D. (1998). Clearance pricing and inventory policies for retail chains. *Management Science*, 44(3):285–300.

[Su, 2007] Su, X. (2007). Intertemporal pricing with strategic customer behavior. *Management Science*, 53(5):726–741.

[Talluri and Van Ryzin, 2006] Talluri, K. T. and Van Ryzin, G. J. (2006). *The theory and practice of revenue management*, volume 68. Springer Science & Business Media.

[Wang, 2016] Wang, Z. (2016). Intertemporal price discrimination via reference price effects. *Operations research*, 64(2):290–296.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Not Applicable]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Not Applicable]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Not Applicable]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]