Nonparametric Bandits Leveraging Informational Externalities to Learn the Demand Curve

Ian N. Weaver
National University of Singapore Business School; i.weaver@nus.edu.sg

Vineet Kumar
Yale School of Management; vineet.kumar@yale.edu

Lalit Jain
Foster School of Business, University of Washington; lalitj@uw.edu

We propose a novel theory-based approach to the reinforcement learning problem of maximizing profits when faced with an unknown demand curve. Our method is based on multi-armed bandits, which are a collection of minimal assumption nonparametric models that balance exploration and exploitation for maximizing rewards while learning across arms. We leverage the informational externality inherent in price experimentation across arms (price levels) in two ways: correlation between demands corresponds to closer price levels, and demand curves are weakly monotonically decreasing. The first information externality is captured by the use of Gaussian process bandits. We expand on this literature by incorporating the second information externality (monotonicity) into Gaussian process bandits; we provide both a monotonic version of GP-UCB and GP-TS. Incorporating these informational externalities limits unnecessary exploration of certain prices and performs better (characterized by greater profitability or reduced experimentation) than current benchmark approaches. Additionally, our method can be used in conjunction with methods like partial identification. Finally, we provide further variants of these algorithms which account for heteroscedasticity in the noise of the purchase data. Across a wide spectrum of true demand distributions and price sets, our algorithm demonstrated a significant increase in rewards, most notably for underlying WTP distributions where the optimal is low (among the set of prices considered). Our algorithm performed consistently, achieving over 95% of the optimal rewards in every simulation setting tested.

Key words: Multi-armed Bandits, Reinforcement Learning, Pricing

1. Introduction

We propose a new method based on reinforcement learning using multi-armed bandits (MABs) to efficiently learn an unknown demand curve. Our algorithm is a nonparametric method based on Gaussian process Thompson sampling that incorporates microeconomic theory to converge towards optimal pricing and profits with significantly less experimentation than current methods. We use weak restrictions on the monotonicity of demand, which creates an informational externality between the outcomes of arms in the MAB. Our method exploits this informational externality to achieve higher profitability with the same level of experimentation as extant methods. The learned demand curve is also guaranteed to satisfy criteria based on economic theory. The proposed method has several other advantages relative to state-of-the-art approaches, and can also be used in a complementary manner with recent approaches like partial identification (Misra et al. 2019).

In practice, the demand curve is only known in the price range that an established product is sold, and unknown elsewhere. Pricing mistakes are greatest for new products that differ most from past products (Huang et al. 2022). Consider the Atlanta Falcons, who in 2016 announced they would be dramatically slashing concessions to untested prices to improve brand equity. When asked how they projected the volume of sales to change, the CEO of the ownership group of the Falcons, Steve Cannon, replied, "It could be a 10 percent bump, it could be a 30 percent bump, who knows."[1] The next season sales volume for food rose 50%.[2] Even with a sophisticated marketing team, there is no perfect substitute for price experimentation about a particular product. Overall, McKinsey estimates that "30 percent of the thousands of pricing decisions companies make every year fail to deliver the best price."[3] Knowledge of the demand curve is also a primary starting point for managers to implement pricing, promotion, and distribution strategies.

Firms experiment often to learn demand using trial and error (Furman and Simcoe 2015). Even in categories that are well known, demand can undergo significant changes over time. To learn demand at different price points, a simple approach is to use a balanced experiment (also called A/B testing) and randomly allocate consumers across a set of different prices. However, firms across a wide range of industries are reluctant to do much price experimentation (Ariely 2010), as managers worry that price experimentation can confuse or alter customers' expectations for uncertain gains. Managers often do not run experiments long enough, making detecting effects difficult (Hanssens and Pauwels 2016). These factors imply that a method which can identify and learn the critical part of the demand curve efficiently from minimal experimentation can be quite valuable in consequently identifying the optimal price points for products. Pricing differs from advertising, where many companies are willing to experiment (Pfeffer and Sutton 2006, Sahni and Nair

[1] https://www.ajc.com/sports/the-economics-the-falcons-new-cheap-stadium-food/8xvH1bAYTewjU2KQoc5HyN/

[2] https://www.washingtonpost.com/sports/2019/03/06/were-evangelists-this-why-atlanta-falcons-are-selling-hot-dogs/

[3] https://www.mckinsey.com/business-functions/growth-marketing-and-sales/our-insights/using-big-data-to-make-better-pricing-decisions

2020, Huang et al. 2018, Simester et al. 2009). The pricing problem is particularly suited to MABs which simultaneously learn while earning, avoiding the wasteful explorations using A/B testing. They deal with the classic trade-off between *exploitation* (current payoff) and *exploration* (learning additional information) as the agent tries to maximize their rewards over some horizon.

The core ideas in this paper expand on canonical bandits by leveraging two distinct but related sources of *informational externalities* across arms. First, we know that demand at closer price points is more likely to be similar compared to demand at more distant price points. This feature of the pricing problem, which we term an information externality, implies that knowing the demand at a focal price point helps in learning about demand not only at that price, but also at other price points. Such learning is more likely to be greater for prices close to the focal price, and less for prices further away from it. The second informational externality is the characterization from microeconomic theory that aggregate demand curves are monotonic. Thus, the quantity demanded at a focal price must be *weakly lower* than the demand at all prices lower than the focal price. Incorporating these related sources of informational externality potentially helps us learn demand more efficiently and accurately, and forms the focus of this paper. We also explore whether the two informational externalities are substitutes, or whether they can act together in a complementary manner to improve performance.

The primary advantage of our method is efficiency in learning to price optimally in a flexible nonparametric manner when the firm has no prior information about demand. If a firm is willing to undertake adaptive experimentation to learn demand, but faces a high cost of experimentation it wishes to minimize, our method helps the firm achieve a higher profitability than current approaches across a robust range of underlying distributions of consumer valuation (willingness-to-pay). A second advantage is that the method offers a guaranteed way to incorporate theoretical knowledge into reinforcement learning problems. It can be applied to any vertical quality-like attribute, not just prices. More broadly, any domain-specific restrictions that can be imposed based on prior conceptual knowledge can be incorporated into this framework. Third, the method provides an estimate of uncertainty, along with point estimates. In fact, the entire posterior distribution can be obtained to provide a complete characterization of uncertainty around the learned demand curve. Fourth, there is little to no human judgment required. Unlike most typical RL models, hyperparameter tuning is automatic, allowing the bandit to run in real time. Fifth, an important aspect is that we do not require any knowledge about the market or consumer characteristics, unlike partial identification approaches like UCB-PI, which requires some knowledge of a consumer's segment membership (Misra et al. 2019). Finally, the method can be used in conjunction with partial identification approaches.

Our method uses nonparametric reinforcement learning (RL) with simultaneous learning and earning, and is most appropriately situated in the class of other nonparametric RL models. The Upper Confidence Bound (UCB) and Thompson sampling (TS) form the underlying set of models that are commonly used for RL. The UCB algorithm (Auer 2002, Auer et al. 2002) is designed to explore the arms with higher payoffs, but also arms that have been less explored, with the idea that greater uncertainty implies greater potential rewards. The TS approach involves a Bayesian updating of the rewards distribution corresponding to each of the arms as they are played (Thompson 1933). Arms are chosen probabilistically, with arms that have a higher mean more likely to be chosen. TS is a stochastic approach, whereas UCB is deterministic.

Gaussian processes (GP) have been used with both UCB and TS classes of RL algorithms. These first model a GP on the data (arm, rewards) and provide the ability to learn a more general reward function, as compared to learning about rewards corresponding to specific arms. A major challenge in using GPs for pricing problems is that we may obtain non-monotonic demand functions that are not consistent with economic theory. This issue is exacerbated by the fact that a GP models the entire demand distribution, meaning particularly noisy data at one price could affect the demand estimation at another. Therefore, a method that can obtain monotonic demand curves from a GP is highly valuable; however, specifying monotonicity in a GP is not trivial.

We propose a model that uses critical properties from economic theory to guide the reinforcement learning process, by enforcing monotonicity on nonparametric bandits. More generally, this method can be easily adapted to any bandit situation where the underlying data has a vertical attribute. Our method provides a practical, quick, implementable algorithm with minimal assumptions or human judgment, which outperforms a wide range of benchmarks. The method is nonparametric, implying it does not depend on a restrictive (parametrized) model of what a demand curve should look like. We specify monotonicity by building up a function interpolating its value from the sum of its derivatives at nearby points. Our method relies on the crucial foundation that the derivative of a GP is also a GP. For decreasing functions, the first derivative is always negative at every point. We leverage the idea that sign restrictions are easy to impose relative to shape restrictions like monotonicity, and we are able to then impose sign restrictions in a different GP space (the space of derivative GPs). Thus, we can restrict the first derivative of demand to be negative, which leads to our method drawing only monotonic demand curves from the GP.

There are two main contributions. To researchers and practitioners, we provide a method that builds upon Gaussian process bandits to account for economic theory imposing the general property that demands curves are downward sloping. We specify an algorithm that efficiently obtains only monotonic, downward-sloping demand curves throughout the experimentation process. Our approach results in significantly higher profits relative to state-of-the-art methods in the MAB literature while having a lower variance between trials. The increase in efficiency means that a larger number of prices can be included in the consideration set. These are important managerial considerations given the reluctance to do pricing experiments, as compared to (for example) advertising experiments. More broadly, in other situations where data has some general known form a priori, our approach shows how such constraints can be combined with the flexibility of nonparametric bandits to improve empirical performance.

In simulations across a range of settings, our proposed algorithm outperforms a set of benchmarks. We found that our algorithm gained higher expected total rewards than our benchmark algorithms including UCB, TS, GP-UCB, and GP-TS. Averaged across three main simulation settings, we found that incorporating monotonicity into GP-UCB and GP-TS consistently increased rewards by 10-26% after 500 consumers and 4-8% after 2500 consumers regardless of the price set granularity. There was, however, massive heterogeneity in the results depending on the underlying distribution with the biggest boost occurring for distributions where the optimal price was low within the price set under consideration. This is because benchmarks tend to over-explore higher prices as a result of the increasing scaling factor of the potential reward as the price increases. Meanwhile, our algorithm leverages monotonicity to limit over-exploration of higher prices, leading to more consistent results: over 95% of the optimal rewards in every scenario tested.

2. Literature Review

The research here is related to several streams, which we detail below.

Learning Demand: Most studies in marketing, economics, and operations make strong assumptions about the information that a firm has regarding product demand. The strongest assumption used is that the firm can make pricing decisions based on knowing the demand curve (or WTP) (Oren et al. 1982, Rao and Bass 1985, Tirole 1988). A generalization of this assumption is that firms know the demand only up to a parameter, used in some of the earlier works on learning demand through price experimentation (Aghion et al. 1991, Rothschild 1974). Typically, in marketing, a consumer utility function is specified in terms of product characteristics, price, and advertising, and preference parameters are estimated from data (Zhang and Chung 2020, Jindal et al. 2020, Huang et al. 2022). However, a robust pricing policy must be able to incorporate all possible demand curves, making a nonparametric approach the gold standard. Nonparametric approaches have been used to account for state changes between periods, but often require simplifications to ensure analytical tractability (Bergemann and Schlag 2008, Handel and Misra 2015).

Closely related to our dynamic pricing experimentation setting, nonparametric approaches in the operations literature have modeled an *exploration* phase followed by an *exploitation* phase (Besbes and Zeevi 2009). While the algorithm in Besbes and Zeevi (2009) corresponds to a balanced experiment, further refinements to the “learn, then earn” approach have been made (Cheung et al. 2017, Wang and Hu 2014). For targeted dynamic pricing and promotions, machine learning methods like hidden Markov models (Zhang et al. 2014) and reinforcement learning (Liu 2022) have been used. Of particular interest, Dubé and Misra (2023) is one of the few actual pricing experiments seen in the field.

Multi-armed Bandits: Multi-armed bandit (MAB) methods are an active learning approach based on Reinforcement Learning and are used across many fields, with business applications in advertising (Schwartz et al. 2017), website optimization (Hill et al. 2017, Hauser et al. 2009), recommendation systems (Kawale et al. 2015), and arcade games (Osband et al. 2016).

Two fundamental bandit algorithms are Upper Confidence Bounds (UCB), based on Auer et al. (2002) and Thompson sampling (TS), based on Thompson (1933). Notably, UCB and TS bandits are shown to outperform “learn, then earn” over a fixed number of trials even when the ex-post optimal exploration time in “learn, then earn” was used (Ferreira et al. 2018, Misra et al. 2019). However, bandits are more flexible than that, which is important for practical purposes, as companies may only be able to commit to limited experimenting before choosing a fixed price (Cheung et al. 2017). There are several bandit models based on TS and UCB methods. For example, Urteaga and Wiggins (2022) use a Bayesian nonparametric Gaussian mixture model to flexibly estimate the reward density (rather than specify a parametric form) in combination with TS.

Relevant to our problem of pricing to learn the demand curve, we have some cognate studies. Bastani et al. (2022) use TS to learn the shared structure across a sequence of dynamic pricing experiments for related products. One particularly notable contribution incorporating monotonicity into bandits is the partial identification method (Misra et al. 2019). Partial identification formalizes the notion that the rewards from a specific arm (price) can be dominated by another arm. Critically, this method relies on the availability of highly informative segmentation data to obtain demand bounds for the segment. The demand bounds are then aggregated across all segments to obtain the corresponding aggregated rewards bounds for each price. Dominated prices can then be eliminated if the lower bound for one price is higher than the upper bound for a different price. In contrast, our algorithm uses the assumption of a weakly decreasing demand curve at the aggregate level, and it does not need informative segmentation to leverage the gains from the monotonicity assumption. As the mechanism is different, it is possible to use partial identification as a complement to our approach. Overall, one limitation of TS and UCB in pricing is that they consider rewards to be independent between arms, ignoring the information from the underlying demand curve.

Gaussian Processes: Gaussian processes (GP) (Williams and Rasmussen 2006) flexibly model distributions of functions, with one of the main advantages being that they not only predict means, but also the uncertainties surrounding these estimates. GP bandits allow for dependencies across arms and were introduced by Srinivas et al. (2009) through the *GP-UCB* algorithm, which fits a GP to the data and then picks a test point using a UCB-like decision rule based on the estimated mean and variance at each test point.[4]

One difficulty with GP-UCB and related approaches is that performance is dependent on human judgment for choosing an acquisition function, which balances the trade-off between exploration and exploitation. The performance of each acquisition function varies

[4] Other bandit algorithms have been created to allow for an unknown function correlating arms including Lipschitz bandits (Bubeck et al. 2011, Kleinberg et al. 2008), Asymptotic Randomized Control bandits (Cohen and Treetanthiploet 2021), and Gaussian process bandits (Srinivas et al. 2009, Chowdhury and Gopalan 2017, Ringbeck and Huchzermeier 2019).

greatly, dependent on the underlying distribution (Hoffman et al. 2011), which is not known a priori. An alternative approach is to use GP-TS (Chowdhury and Gopalan 2017), which combines the GP estimation with TS instead of UCB, or their Improved GP-UCB algorithm.

Another noteworthy related contribution is by Ringbeck and Huchzermeier (2019), who model a pricing problem using GP-TS. This paper is closely related to the present one, and examines the problem of using GPs to learn rewards across arms for pricing. The GP is modeled here at a demand level, which is important for two reasons. First, it allows them to model demand-level inventory constraints in a multi-product setting, and second, it allows for a separation of the learning problem (at demand level) from the rewards optimization (reward is the product of demand and price).

Our paper builds on this work by incorporating the weakly decreasing nature of demand curves, by providing a principled way to sample Gaussian process bandits so that only weakly decreasing estimates are obtained. We provide monotonic variants for both GP-UCB and GP-TS, and more broadly, our approach provides a way to leverage theory-based models incorporated into a reinforcement learning algorithm.

3. Overview of MAB Model

The multi-armed bandit model operates by active experimentation through earning while learning with the objective of obtaining higher cumulative rewards (or equivalently minimize regret). Consumers arrive in sequence, and are presented a price chosen by the MAB algorithm. The consumer then decides whether to purchase or not, based on whether the offered price is lesser or greater than their valuation, or willingness to pay (WTP). Consumers are heterogeneous in WTP. The population of consumers thus has a (population level) distribution of WTP, which corresponds directly with a demand curve.

There are a number of assumptions that are needed for the model. First, consumers are drawn randomly from a population that has a stable WTP distribution of valuations. Second, consumers are short-lived, and the overall distribution of price expectations does not change based on the experimentation. These sets of assumptions are typical to any field experiment, and necessary for the results of an experiment to be applicable after the experiment has ended.[5]

[5] This rules out dynamic situations where customers may change over time or customers' current decisions are greatly affected by future beliefs. Such situations include strategic consumers (Nair 2007), learning (Erdem and Keane 1996,

Third, the firm is a single-product monopolist that is seeking to maximize cumulative profits under uncertainty, where the demand curve is unknown.[6] Following the literature, the firm is interested in setting a single optimal price, i.e., it does not price discriminate and does not have inventory or other considerations (Besbes and Zeevi 2009, Ferreira et al. 2018, Ringbeck and Huchzermeier 2019, Misra et al. 2019).[7]

3.1. Model Components

The MAB problem applied to pricing has three components: actions, rewards, and a policy.

The first component – *actions* – refers to the set of prices from which the firm can choose. Prior to the experiment, the firm selects a finite set of K ordered prices $P = \{p_1, ..., p_K\}$ where $p_1 < p_2 < ... < p_K$, where prices are scaled so that $0 \leq p_k \leq 1$.[8] While this paper does not explicitly model how to choose the set of prices, the general trade-off is that learning is easier with fewer prices, but a higher optimal is possible when more prices are considered. The results section provides general guidelines for how to pick the set of prices. Once P is chosen, at each time-step t of the experiment, the firm chooses a price p_k from P. Here, we are focused on an environment where the rewards distribution is stationary. Later, we allow for a non-stationary distribution by modeling time-varying demand in Appendix EC.9.

The second component – *rewards* – refers to the profits that a firm makes at each purchase opportunity. The firm faces an unknown true demand $D(p)$, and the true profit function is given by $\pi(p) = pD(p)$. We assume variable costs are zero, although the model can easily accommodate such costs. The true profit is not observed, and instead the firm observes noisy realizations of profits corresponding to each price p_k. Considering the data at each price separately, we define n_{kt} to be the number of times that price p_k (arm k) has been chosen through time t, and s_{kt} to be the cumulative number of purchases for an action k through time t. The sample purchase rate through time t or price p_k is simply $y_{kt} = \frac{s_{kt}}{n_{kt}}$,

Yu et al. 2016), and stockpiling (Ching and Osborne 2020, Hendel and Nevo 2006). These assumptions are typically implicit in most field experiments, e.g., in the advertising literature (Hoban and Bucklin 2015, Lambrecht et al. 2018, Gordon et al. 2019). For example, if strategic consumers seeing a discount believe the company is experimenting and might discount even more at a later time, then the results would not accurately reflect the treatment effect.

[6] This assumption can actually be weakened, as results will hold as long as the algorithm is deployed in a stable environment where competitors do not change prices strategically in response to real-time changes and firms are not worried about future competitive entries (Rubel 2013).

[7] Inventory constraints are necessary when inventory is limited, such as in clothing. When the product is limited, it may be best to forgo selling to a customer in lieu of another customer with a higher WTP. For products without production constraints (e.g., a Netflix subscription) this is not an issue.

[8] With unscaled prices $\{\widetilde{p_1}, ..., \widetilde{p_K}\}$, the set of scaled prices can be created by dividing any price by the largest price in the set, i.e., $p_k = \widetilde{p_k}/\widetilde{p_K}$.

which is consistent with the standard notation for Gaussian processes. Accordingly, the sample profit $\bar{\pi}_{kt}$ for a price p_k at time t is $\bar{\pi}_{kt} = p_k(\frac{s_{kt}}{n_{kt}})$.

The final component – a *policy* – denoted by Ψ, is a decision-making rule that picks an action or price in each round using the history from the past rounds. In this situation, the history can be written as $H_t = \{S_t = (s_{1t}, ..., s_{Kt}), N_t = (n_{1t}, ..., n_{Kt})\}$. Formally, in round t, the policy picks a price using the history from the past $(t-1)$ rounds: $p_{k_t} = \Psi(P, H_{t-1})$. What distinguishes various MAB algorithms is how this policy is defined. For a typical randomized experiment, the policy can be defined as an equal probability across all arms, completely ignoring history.

To summarize, there is an unknown, but fixed distribution of consumer valuations for a product. In each round, a consumer interacts with the firm, which shows a price, and then the consumer makes a purchase decision, which is then observed by the firm. Following Misra et al. (2019), we focus on discrete choice purchases (the consumer buys zero or one units of a product) and the company is only able to change the price every 10 consumers.[9] A summary of the notation is given in Table 1.

Table 1 Summary of Bandit Notation

Notation	Description	Formula
Ψ	Policy for dynamic pricing (i.e., decision rule)	Depends on algorithm
k	Action index: the set of K actions	$k \in \{1, 2, ..., K\}$
t	Time-step: denotes the t-th customer of the price experiment	
n_{kt}	Number of times price p_k has been chosen through time t	
s_{kt}	Number of purchases at price p_k through time t	
H_t	History from past t rounds of experiment	$H_t = \{S_t = (s_{1t}, ..., s_{Kt}), N_t = (n_{1t}, ..., n_{Kt})\}$
k_t	Action chosen at time t: this is dependent on the set of actions, policy and past history	$k_t = \Psi(P, H_{t-1})$
p_k	Scaled prices	$p_k \in P = \{p_1, ..., p_K\}$ where $0 \leq p_k \leq 1$ $\forall p_k$
$D(p_k)$	Demand at price p_k	
π_{k_t}	Profit realized when price p_k was tested in time period t	
y_{kt}	Mean demand through time t of price p_k	$y_{kt} = s_{kt}/n_{kt}$
$\bar{\pi}_{kt}$	Sample mean profit through time t of price p_k	$\bar{\pi}_{kt} = p_{kt}(s_{kt}/n_{kt})$

[9] Alternatively, one could specify that prices can be changed every X minutes and have customers arrive according to a Poisson distribution. Both setups are just approximations of how the algorithm may be deployed in a practical application.

3.2. Policies: Algorithms for Choosing the Arm to Play

To assess the empirical performance of our proposed algorithm, we need to compare its performance to other various algorithms. Note that in Table 1 only Ψ – the policy – depends on the algorithm, while all other elements in the table are *independent* of the algorithm. Regardless of other modeling choices, a necessary component of all bandits is a specification for choosing the arm given the history. Thus, the empirical performance of different algorithms will be measured by comparing the average rewards per consumer over numerous simulations, as well as the variance of these rewards. Algorithms having higher average total rewards and lower variance would be achieving higher performance.

3.3. Baseline Policies – Deterministic and Stochastic Algorithms

The simplest algorithm or policy – a randomized experiment or A/B testing approach – would just select a random arm and completely ignore the history of arms played and corresponding outcomes. Next is the class of myopic policies, such as greedy-based algorithms (i.e., ϵ-greedy) and softmax (Dann et al. 2022).

Finally, the more sophisticated baseline algorithms that we build upon are broadly in two separate classes. The first class consists of algorithms based on Upper Confidence Bound (UCB), which balances the observed rewards across arms with the uncertainty. UCB-based policies account for the fact that there is more potential uncertainty for arms that are played less frequently. The second class of algorithms is based on Thompson sampling (TS), and directly quantifies the uncertainty using a Bayesian approach. Prior research has provided provable guarantees on regret for these algorithms. We refer to these as our baselines, since we develop our method based on these two classes of policies. Technical details about UCB and TS can be found in Appendix EC.1.

UCB: The Upper Confidence Bound algorithm is a deterministic nonparametric approach popularized because it is proven to asymptotically have the best possible performance in terms of achieving the lowest maximum regret (Lai and Robbins 1985, Agrawal 1995, Auer et al. 2002). Regret is defined as the difference between the cumulative reward of the optimal strategy and the cumulative reward obtained by the chosen strategy.

In every round, UCB uses a formula that scores each action, and then the highest scoring action is picked. As it is deterministic, it will always choose the same action given a particular data history. The scoring rule is the sum of an exploitation and exploration term. The exploitation term is the sample mean of past rewards at a given action, which informs which actions have previously had higher payoffs. A fully exploitative strategy[10] would use just this first term. The exploration bonus, meanwhile, is increasing in how uncertain we are about the sample mean for an action; that is, it decreases with the number of times an action has been chosen. Thus, the UCB algorithm balances between a mix of exploitation and exploration. Notably, in the pricing setting, we scale the exploration term by price as in Misra et al. (2019).

TS: Thompson sampling is a randomized Bayesian parametric approach. For each action, a reward distribution is specified a priori and updated based on the history of past trials (Thompson 1933). In each round, an action is chosen according to the probability that it is optimal given the history of past trials. The easiest implementation is to take a sample from each distribution during each round and pick the arm that gives the highest payoff from these samples. In our setting, where purchases are either 0 or 1, the TS approach is operationalized by sampling from a scaled beta distribution where the parameters are the number of successes and number of failures. That is, at time $t+1$ for a given arm k, a sample is taken from $\text{Beta}(s_{kt}+1, n_{kt}-s_{kt}+1)$ and then scaled by the price, p_{kt}; the arm with the highest value is then chosen.

A key assumption of Thompson sampling is having a suitable parametric form for the distribution. In addition to TS, we also look at Nonparametric Thompson sampling, or *N-TS*, by Urteaga and Wiggins (2022), which flexibly estimates the reward density with nonparametric Gaussian mixture models.

Performance Metrics: Multi-armed bandits are designed for learning while earning. There are a number of performance metrics that are used to evaluate the performance of policies. Among the most common is regret, which characterizes the difference in rewards (profits in our setting) between the arms that were played and the rewards obtained by only playing the optimal arm. Among a set of algorithms, the one with the lowest cumulative regret is the same as the one with the highest cumulative rewards. Formulating the bandit problem as a statistical problem (regret) rather than an optimization problem (maximizing cumulative reward) lends itself better to theoretical guarantees (Cohen and Treetanthiploet 2020). Theoretical guarantees often state that an algorithm has the lowest possible bound for expected regret, however, this is subtly different from empirical performance, which

[10] This would be equivalent to a fully greedy algorithm.

may be higher for algorithms without such theoretical properties.[11] Another important criterion for businesses (especially those with few products who are unlikely to run many price experiments) is the variance in the rewards between simulations. A third could be to evaluate consistency with microeconomic theory, e.g., whether the demand curve learned by the algorithm satisfies conditions like monotonicity. A full description of the performance metrics is discussed in Appendix EC.2.

3.4. Leveraging Informational Externalities

Our goal is to create a general method where any decision rule (either UCB, TS, or any other rule) can be combined with the two informational externalities that are relevant to the problem of learning the demand curve in order to determine optimal pricing. We start by noting that the baseline UCB and TS algorithms do not feature any information externalities, i.e., the arms are viewed as being independent of one another. Learning about the rewards for one arm (price) will not alter the expected rewards for any other arm, which is the issue that we seek to model with informational externalities.

First Informational Externality: The first information externalities recognize local dependence of functions through continuity. In the pricing application, specifically, we know that demands corresponding to a pair of prices tend to be closer together when the prices are closer together. To inform demand at arm j or (price) p_j, the demand at p_{j-1} and p_{j+1} are most informative. More generally, it is possible to learn about the demand $D(p)$ at price p, from observed demands at nearby price points, say $D(p+\epsilon)$ for small ϵ. The information spillover generated is symmetric and bidirectional, and points that are further away are less important, and given less weight due to the structure of the covariance matrix.

Second Informational Externality: The second informational externality specified by the monotonicity property has a more global meaning, since demand at price p_j constrains all the demands at higher prices, since $D(p_k) \leq D(p_j)$ when $p_k \geq p_j$. The impact is also asymmetric. Specifically, we know that demand at a higher price p_k is upper-bounded by demand at a lower price p_j, and demand at a lower price is lower-bounded by demand at a higher price. If a demand curve is specified, monotonicity must be satisfied at each point of the demand curve, and not just at the chosen price levels corresponding to the arms.

[11] For example, it is proven that under certain conditions UCB has the lowest possible bound for expected regret (Auer et al. 2002). However, empirically, under the same conditions it is often outperformed by greedy algorithms with respect to maximizing rewards (Bayati et al. 2020).

From Points to Functions: The logic of sharing information between arms leads to the idea of using functions instead of rewards at specific arms (price points). The approach of using functions that span the support of the arms (price levels) serves as a foundation to incorporating and modeling the two informational externalities. Without functions, the approach to including dependence across arms would necessarily be ad hoc and pre-specified by the researcher. Using functions works in this case since we know from microeconomic theory that the arms are not quite independent. For example, knowing the demand at price $p = 0.5$ is likely to be informative of demand at $p = 0.6$. We can specify certain functional forms (e.g., splines) that flexibly model the true demand curve. However, the risk is that any parametric approximation chosen by the researcher might be insufficient to capture the true shape of the demand curve.

From Functions to Gaussian Processes: We use the idea of modeling the demand curve (or function) as a draw from a Gaussian process (GP). Gaussian processes (GPs) have been used across a wide range of machine learning applications, including bandits (Srinivas et al. 2009), due to their flexible nonparametric nature. GPs are nonparametric since they allow any function to be drawn probabilistically from the set of functions on the chosen support, unlike most commonly used methods in economics and marketing. Rather, any arbitrary demand curve can be modeled, and the GP learns about the shape from the data. This flexible approach can be used to incorporate dependencies across arms.

The GP-based approach incorporates both of the informational externalities. The primary algorithms used in this paper are Gaussian process bandits adapted to dynamic pricing. GPs have recently been used in multi-armed bandits by combining the GP approach along with a decision rule like UCB or Thompson sampling (TS) by Chowdhury and Gopalan (2017), who evaluate the theoretical and empirical performance of GP-UCB and GP-TS. More specifically to the pricing problem, GP-TS has been used in the evaluation of pricing in a multi-product setting by Ringbeck and Huchzermeier (2019).

3.4.1. Advantages of GPs relative to other methods GPs have many desirable features for the present class of problems. First, GPs provide both a parsimonious and nonparametric way to incorporate both informational externalities in a transparent, principled, and provable manner.[12] Second, GPs also have closed form solutions where hyperparameters can be turned very quickly with maximum likelihood estimation. Intuitively, however,

[12] One alternative way would be to use a parametric model like *GLM-UCB* (Filippi et al. 2010) and restrict the coefficients to obtain weakly decreasing demand functions. Given we model demand on just one variable, price, a nonparametric method is much more flexible and less likely to be negatively impacted by model misspecification.

they work well with bandits because the exploration-exploitation trade-off depends on having knowledge of the distribution of the rewards, not just the mean reward at each arm.[13] A GP provides a probability distribution of functions, offering a principled way of obtaining exactly what is needed to manage the exploration-exploitation trade-off.

Finally, we note that other nonparametric methods (including many machine learning methods), may possibly provide higher accuracy estimates for the mean, but because they are not as competent at knowing the certainty to which the prediction is true, they are ill-suited for bandit algorithms where controlling the exploration-exploitation trade-off is crucial. Meanwhile, GPs can quantify what they do not know, which allows the uncertainty around the estimates to be incorporated into the decision-making process in a principled manner. This is why a method like GPs is needed where the entire distribution of functions is estimated when creating a smoothing alternative to the raw data. In fact, one can consider the case where just the means of the posterior GP are used rather than using the entire posterior GP.[14] This was tested in Srinivas et al. (2009), who found it was "too greedy too soon and tends to get stuck in shallow local optima" (p. 4). That is, it can cause the algorithm to get stuck and under explore, leading to inaccurate results.

4. GP Bandits Model

We now provide the complete specification for our model. The key building block to our approach is the Gaussian process. A Gaussian process (GP) is a stochastic process (collection of random variables) such that every finite subset of random variables has a multivariate Gaussian distribution. It can be thought of as a *probability distribution over possible functions*. In our setting, this would mean that a draw from the GP would represent a demand curve. A complete overview of GPs can be found in Appendix EC.3.

Gaussian processes can be combined with bandits such as UCB (Srinivas et al. 2009) and TS (Chowdhury and Gopalan 2017, Ringbeck and Huchzermeier 2019). The main difference is that instead of using the raw data directly, a posterior GP is estimated before using a UCB scoring rule or Thompson sampling. The advantage is that a GP estimates the entire function over the support, when raw data does not use information from nearby arms. This

[13] This is also incorporated in the UCB algorithm, which models both a term for the sample mean of each arm as well as an exploration bonus dependent on the bounds surrounding those sample means. However, the interpretation of the bonus term in terms of uncertainty is less clear.

[14] One can think of this as a greedy-based GP algorithm.

can minimize incorrect "exploitation" in early rounds, while still learning the true rewards over the long run. In this paper, both UCB and TS are adapted to the pricing scenario with Gaussian processes. We seek to remain fairly agnostic between GP-UCB and GP-TS, as past literature (Chowdhury and Gopalan 2017) corroborated by our own simulations shows that neither completely dominates the other. The goal is to determine whether and under what conditions incorporating monotonicity into the GP improves performance for either algorithm.

We next define the additional notation required for GPs, explain the role of hyper-parameters and how they are obtained (for shape and noise), and then specify how the theoretical restriction of monotonicity of demand curves is incorporated into the bandit setting. Notably, in our setting, input data is obtained only at the test prices, allowing for the size of the input data to remain fixed as the number of observations increases. We denote the input training prices as P_t and the corresponding sample mean demands as $\mathbf{y} = (y_{1t}, y_{2t}, ..., y_{Kt})$. Both of these will have a maximum size $|P|$ once every price in the test set P has been tried at least once. As in Ringbeck and Huchzermeier (2019), we model the GP at the demand-level and then scale by price so that the arm decision is at the reward-level.

Table 2 Summary of GP Notation

Description	Notation in Pricing Setting
Training data noise	$\boldsymbol{\sigma_y^2}$
RBF kernel	$k^{\text{RBF}}(p_i, p_j) = \sigma_f^2 e^{\frac{-(p_i - p_j)^2}{2l^2}}$
RBF hyperparameters	$\{\sigma_f, l\}$
Covariance function (kernel) evaluated at two points	$k(p_i, p_j)$
Covariance matrix between price vectors	$K(P, P)$

Kernel: A Gaussian process flexibly models the dependency across input points using a kernel, which is implemented by using a covariance function. A kernel $k(\cdot,\cdot)$ takes in two points (in our setting, prices p_i and p_j) from the input space and returns a scalar representing the covariance between the outputs at those points. From this, a covariance matrix $K(\cdot,\cdot)$ can be created for a set of inputs. A common and robust kernel for GPs is the radial basis function (RBF) kernel, also known as a Gaussian kernel or squared exponential kernel (Duvenaud 2014).

$$k^{\text{RBF}}(p_i, p_j) = \sigma_f^2 e^{-\frac{(p_i - p_j)^2}{2l^2}} \tag{1}$$

The above kernel has a set of desirable properties that are necessary for our setting. It is differentiable and can provably approximate an arbitrary continuous target function uniformly on any compact subset of the input space (Micchelli et al. 2006). It has two shape hyperparameters, σ_f and l. The first hyperparameter, σ_f, is a scale factor that controls the amplitude of the function (i.e., the average distance the function is away from its mean). The second hyperparameter, l, determines the smoothness[15] of the function, which means that it describes how the correlation between two points drops as the distance between them increases (Shahriari et al. 2015).

Hyperparameters: As the kernel controls the function shape, a crucial practical step in GP bandits implementation is the selection of the kernel hyperparameters. To minimize human (researcher) judgment, it would be ideal to have an efficient, accurate, and automatic method for selecting hyperparameters. There are two types of hyperparameters that need to be specified: shape parameters for the kernel, (σ_f, l), and the noise parameter, σ_y^2, which dictates how noisy the data is.

There are two main non-Bayesian methods for tuning the hyperparameters.[16] The first method is the typical machine learning method of estimating the GP on training data and testing a grid of hyperparameters to see which performs best on the test data. The second standard approach is to choose values of the hyperparameters that maximize the likelihood of the data given a model. Mathematically, this is equivalent to minimizing the negative log marginal likelihood (equation 2.30 in Williams and Rasmussen (2006)).

$$\log \text{prob}(\boldsymbol{y}|P_t) = -\frac{1}{2}\boldsymbol{y}^T(K(P_t,P_t)+\boldsymbol{\sigma}_{\boldsymbol{y}}^2 I)^{-1}\boldsymbol{y} - \frac{1}{2}\log|K(P_t,P_t)+\boldsymbol{\sigma}_{\boldsymbol{y}}^2 I)| - \frac{t}{2}\log(2\pi) \qquad (2)$$

where t denotes the number of data observations (i.e., the number of rounds in a bandit).

Noise Parameters: We specify the noise parameters directly because it can be difficult to disentangle shape and noise parameters if both are estimated together (Murray 2008).[17] We use the fact that purchase decisions are binary in our model to characterize the upper bound of the variance at any price. When prices are set in $p \in [0,1]$, the variance for a

[15] Intuitively, this can be thought of as the length of the "wiggles."

[16] Bayesian tuning methods are often not too slow and not suitable for real-time bandit settings.

[17] For example, consider two close input points (x-axis) that have very different outputs (y-axis). One possible explanation is that the data is accurate and the GP needs shape parameters that permit sufficiently high variation to allow for large output differences from nearby inputs. Alternatively, it could be that the true outputs are actually close together but that the data is very noisy; in this case, the previous shape parameters would be overfitting.

Bernoulli random variable is $p(1-p)$, which implies that the maximum variance in the purchase probability is 0.25 that occurs when the true purchase probability is $p=0.5$. All other prices would have a lower variance. Thus, a conservative approach without estimating the noise parameter would be to set the noise variance to 0.25. While this might be too conservative for some prices, using noise parameters that are too small could lead to a lack of search and potentially poor results from getting stuck in errant equilibria. A discussion of alternative methods where we allow the noise to take on different values for different prices (that is, heteroscedastic noise) is discussed in Appendix EC.10.

Computational Complexity: We note here that the complexity grows in the number of arms, not the size of data (or number of customers). Please see Appendix EC.4 for details.

4.1. GP-UCB and GP-TS Algorithms

We can now define the GP-UCB and GP-TS algorithms in the pricing scenario. To initialize the algorithm, we choose the first price randomly.[18] Once one data point becomes available, the hyperparameters are chosen using equation (2). At time t, using price set P and $\mathbf{y}=(y_{1t},y_{2t},...,y_{Kt})$ the posterior mean, $\mu(D^*)$, and covariance matrix, $\mathrm{Cov}(D^*)$, can be calculated by equations (EC.9) and (EC.10).[19]

With the GP estimated, UCB and TS can be applied. For GP-UCB, the posterior demand mean, $\mu_t(p_k)$ and its posterior variance $\sigma_t^2(p_k)$ at each price $p_k \in P$ are used to decide the price at iteration $t+1$ by using the following decision rule:

$$p_k^{\text{GP-UCB}} = \underset{p_k \in P}{\arg\max} \left(p_k(\mu_t(p_k) + \beta_{t+1}^{1/2}\sigma_t(p_k)) \right) \tag{3}$$

where $\beta_t = \frac{2}{5} log(|P|t^2\pi^2/(6\delta))$ and δ is set to 0.1.[20] Note we have scaled by p_k as the algorithm is optimizing on reward (rather than at the demand level where the GP was estimated).

On the other hand, in GP-TS, instead of having to sample at each arm like in TS, a demand draw for every test price, $d_t(p_k)$ can be obtained by sampling from the posterior GP. That is, $d_t(p_k)$ is a sample taken from the posterior normal distribution with given

[18] Another possible initialization method is to set arbitrary hyperparameters and model the GP without data. Both methods are practical with only small differences in overall performance (under 1% in all our simulations) with neither method dominating.

[19] D^* is a random variable denoting the Gaussian process posterior prediction.

[20] This is the value that Srinivas et al. (2009) found empirically worked well. There may be better values of β for other simulations, though determining β without past data is generally challenging (Hoffman et al. 2011).

mean and covariance matrix $D^* \sim N\left(\mu(D^*), \mathrm{Cov}(D^*)\right)$. Then, using Thompson sampling, the price chosen will be

$$p_k^{\text{GP-TS}} = \underset{p_k \in P}{\arg\max}\left(p_k d_t(p_k)\right) \tag{4}$$

4.2. Monotonic Bandits

We now turn our focus to the main contribution of this paper, which is to incorporate monotonicity constraints into Gaussian process bandits. Our goal is to obtain only weakly decreasing functions, consistent with microeconomic theory that demand weakly decreases with price. Recall that the baseline GP allows for any function, and does not impose any restrictions on the shape, so the goal is to impose monotonic shape restrictions. We will call the versions of GP-UCB and GP-TS with monotonicity *GP-UCB-M* and *GP-TS-M*, respectively. For GP-TS-M, we require a way to randomly draw a monotonic function from the set of monotonic functions in the posterior GP. For GP-UCB-M, we require an estimate of the mean and variance from the subset of monotonic demand curves from the posterior; this can be obtained by averaging over many monotonic draws.

To obtain a random monotonic draw, one simple approach is to use rejection sampling and sample the GP until a weakly decreasing draw is obtained. This approach will work to some extent, but there is no guarantee that a weakly decreasing draw will be found expediently. Furthermore, this issue is exacerbated when there are few observations and when there are many test prices. Specifically, since monotonicity must be satisfied locally at each point as well as globally, there is no alternative to checking this condition at all the price levels. In cases with many arms where the sample means are highly non-monotonic, the probability of finding a monotonic draw can become vanishingly slim.

To ensure that a weakly decreasing draw can be obtained from a GP in an expedient manner in all cases, we develop a method from first principles. The idea is that while sampling a monotonic function from a GP is intractable, obtaining a draw from a GP where all the values are negative is a tractable sampling problem (equivalent to sampling from a truncated normal). Specifically, since a decreasing monotonic function can be characterized as a function whose first derivatives are negative at all points, if a link exists between the GP and its derivatives, we can transform the monotonic sampling problem to one that is tractable. We obtain this link by using the useful property that the derivative of a GP is also a GP (and that the RBF kernel we use is infinitely differentiable), allowing for the derivative of a GP to be estimated from our data. We then use the basis functions proposed by Maatouk and Bay (2017) to recover the demand function estimate from a draw of negative derivatives sampled from the derivative of the GP.

Another advantage of this principled approach is that the function is guaranteed to be monotonic not just at the discrete price levels forming the support, but also at any intermediate price where no experimentation is performed. The only assumption needed is that the demand function is differentiable with a continuous derivative.

Basis Functions: The first step is to estimate the demand function using a collection of functions h_j known as the interpolation basis. The demand function will be estimated by linearly interpolating between the function values at knots spaced over the support. Following the notation of Maatouk and Bay (2017), we let $u_j \in [0,1], j = 0,1,...,N$ denote equally spaced knots on $[0,1]$ with spacing $\delta_N = 1/N$ and $u_j = j/N$. The interpolation basis is defined as

$$h_j(p) = h\left(\frac{p-u_j}{\delta_N}\right) \text{ where } h(p) = (1-|p|)\mathbb{1}(p \in [-1,1]) \tag{5}$$

Then, for any continuous function $D:[0,1] \to \mathbb{R}$, the function

$$D_N(\cdot) \approx \sum_{j=0}^{N} D(u_j)h_j(\cdot) \tag{6}$$

approximates D by linearly interpolating between the function values at the knots u_j. One key property of the interpolation basis is that as the gap between the evenly spaced knots becomes infinitesimally small, the distance between the estimation and the true function converges to 0. From this property, the demand function can be written in terms of its intercept, derivatives, and basis functions as shown in Proposition 1. A proof is provided in Appendix EC.5.1.

Proposition 1. *Assuming the demand function $D:[0,1] \to \mathbb{R}$ is differentiable with a continuous derivative (i.e., $D \in C^1([0,1])$), then it can be estimated by its intercept and derivatives by the following equation:*

$$D(p) \approx D(0) + \sum_{j=0}^{N} D'(u_j)\int_0^p h_j(x)dx \tag{7}$$

While this works for all class C^1 functions on the support, we additionally assume that this unknown demand function D is weakly decreasing, meaning that it belongs to a subset $\mathcal{M}$ defined as follows:

$$\mathcal{M} := \{D \in C^1([0,1]) : D'(p) \leq 0, p \in (0,1)\} \tag{8}$$

That is, D belongs to the subset of functions where the derivative is never positive at any value of p.

To summarize, the main changes between the monotonic and non-monotonic version are as follows. When incorporating monotonicity, instead of estimating a GP of the means at the test prices, we estimate a GP of the derivatives at the knots concatenated with the mean at the intercept. Crucially, the intercept and derivatives must be estimated together, but this is possible due to the property for a GP that the joint distribution of values and their derivatives are also a GP (see Appendix EC.5.2 for details on how to calculate the derivative of a GP). Once the posterior GP is estimated, off-the-shelf sampling can be used to acquire a draw where every derivative is non-negative (we use the *TruncatedNormal* package in R (Botev and Belzile 2021)). We denote $\mathcal{M}$ as the subset of monotonically decreasing functions from the posterior GP. Next, equation (7) provides a formula to recover the demand sample d at the desired test prices using just the draw and basis functions h. From this stage, the decision rules for GP-UCB or GP-TS can be used per usual. Formally, the methods are outlined in Algorithms 1 and 2.

Algorithm 1: GP-TS-M

```
1  Set test prices $P$, kernel $k$, noise hyperparameter $\sigma_y^2$, and knots $\mathcal{U}$
2  Compute the integrals of the basis functions at $\mathcal{U}$ using equation (5)
3  Define test points as $\mathcal{U}_0 = \{0\} \cup \mathcal{U}$ (concatenate the intercept)
4  For $t = 1$ pick price randomly, and observe purchase decision
5  Initialize training input $P_t$ and training output $y_t$
6  for $t = 2, 3, \ldots$ do
7      Compute shape hyperparameters $\sigma_f$ and $l$ using equation (2)
8      Compute covariance matrix (equation (EC.7)) using $P_t$ and $\mathcal{U}_0$ with equations (EC.15), (EC.16),
         (EC.17)
9      Estimate posterior GP using equations (EC.9) and (EC.10)
10     Sample randomly from $\mathcal{M}$ at test points $\mathcal{U}_0$
11     Estimate the demand draw $d_t$ at test prices $P$ using equation (7)
12     Play price $p_k = \arg\max_{p_k \in P} (p_k d_t(p_k))$
13     Observe purchase decision
14     Update $P_t$ and $y_t$
15 end
```

Algorithm 2: GP-UCB-M

```
1  Set test prices $P$, kernel $k$, noise hyperparameter $\sigma_y^2$, and knots $\mathcal{U}$
2  Compute the integrals of the basis functions at $\mathcal{U}$ using equation (5)
3  Define test points as $\mathcal{U}_0 = \{0\} \cup \mathcal{U}$ (concatenate the intercept)
4  For $t = 1$ pick price randomly, and observe purchase decision
5  Initialize training input $P_t$ and training output $y_t$
6  for $t = 2, 3, \ldots$ do
7      Compute shape hyperparameters $\sigma_f$ and $l$ using equation (2)
8      Compute covariance matrix (equation (EC.7)) using $P_t$ and $\mathcal{U}_0$ with equations (EC.15), (EC.16),
         (EC.17)
9      Estimate posterior GP using equations (EC.9) and (EC.10)
10     Obtain $N$ samples from $\mathcal{M}$ at test points $\mathcal{U}_0$
11     Estimate the demand draw $d_{t,n}$ at test prices $P$ using equation (7) for each of the $N$ samples
12     Estimate $\mu_t(p_k)$ and $\sigma_t(p_k)$ from the collection of demand draws
13     Play price $p_k = \arg\max_{p_k \in P} \left( p_k(\mu_t(p_k) + \beta_{t+1}^{1/2} \sigma_t(p_k)) \right)$
14     Observe purchase decision
15     Update $P_t$ and $y_t$
16 end
```

Other Monotonic Bandits: There are multiple possible methods to enforce monotonicity with respect to GP-bandits. In Appendix EC.6, we provide another method that jointly estimates the demand function and its derivative.[21] Notably, while this can enforce negative derivatives at the test points, functions can possibly be non-monotonic between test points. However, with sufficient test points, we find that empirically this method performs nearly identically to Algorithms 1 and 2. Additionally, for the joint method, we provide theoretical properties (regret bound), the same as the bounds for GP-TS without monotonicity.

5. Simple Example Illustrating Demand and Profit Learning

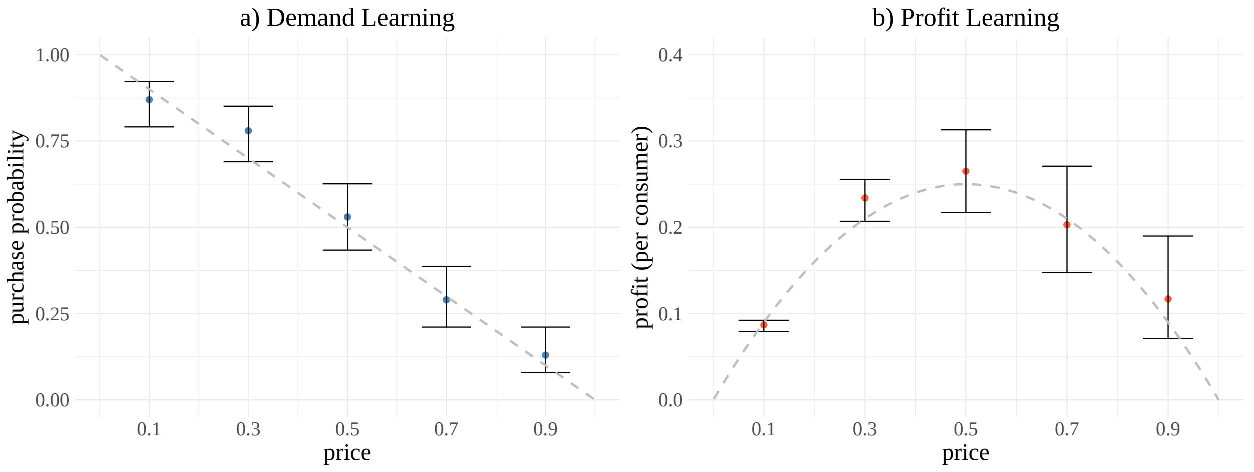
In this section, we discuss a simple example which will illustrate why the bandit setting is different for pricing. This example illustrates the mechanism for why bandit algorithms predictably do worse or better for particular underlying distributions. To illustrate this point, we consider what a balanced experiment would learn during a pricing experiment.

Let us consider a company selling a product where each consumer can only purchase one unit. Let the true unknown demand curve be $D(p) = 1 - p$ and prices tested be $P =$

[21] Again, this uses the properties that the derivative of a GP is also a GP, and that a GP concatenated with its derivative is also a GP.

$\{0.1, 0.3, 0.5, 0.7, 0.9\}$. The corresponding true demands are $\{0.9, 0.7, 0.5, 0.3, 0.1\}$ while the corresponding true profits are $\{0.09, 0.21, 0.25, 0.21, 0.09\}$. The company tests each price 100 times, and the results of the demand learning and profit learning are shown in Figure 1. Figure 1a) shows the sample mean demand at the prices tested (blue dots) along with their corresponding 95% credible intervals,[22] while Figure 1b) shows the sample mean profit at the prices tested (red dots) along with their corresponding 95% credible intervals. The grey dotted lines represent the true demand curve and true profit curve accordingly.

Figure 1 Demand and Profit Learning for a Balanced Experiment



Notes. Results of a single balanced experiment where each of the prices $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ were tested 100 times with true unknown demand of $D(p) = 1 - p$. Figure a) shows the mean and 95% credible intervals for purchase probability at each price tested — the dotted grey line shows the true purchase probability. Figure b) Shows the mean and 95% credible intervals for profit at each price tested — the dotted grey line shows the true expected profit.

This figure illustrates a crucial phenomenon about learning the optimal price. While the credible intervals are roughly the same size at the demand level,[23] they are much different at the profit level. This is because while learning happens at the demand level, profits are obtained by scaling the demand by price, which means that bounds will be larger for higher prices than lower prices. This is apparent from Figure 1b), where the credible intervals increase with the price despite each price having been tested an equal number of times. Intuitively, this makes sense, as being slightly incorrect in an estimate of the demand for a low price has very little effect on the estimate of the profit, while being incorrect on the demand at a high price has a large effect on the estimate of the profit. This leads to large differences in the ability to learn the demand for a certain price. For example, the true

[22] Credible intervals are calculated by assuming a binomial prior.

[23] There are small differences, as discussed in Appendix EC.10.

variance at $p = 0.1$ and $p = 0.9$ are equivalent at the demand level, but at the profit level the interval is nine times larger at $p = 0.9$ than $p = 0.1$.

As bandits make decisions by learning based on expected reward and uncertainty of that measure, they are also susceptible to this scaling issue, as learning is easier at lower prices than higher prices. This leads to a large discrepancy in bandit performance depending on where the optimal price is located among the set of prices being tested. If the optimal price is high, then bandit algorithms perform quite well, as learning that low prices are sub-optimal is not a difficult task. However, if the optimal price is low, it will take a lot of samples to properly learn that the high prices are sub-optimal, which will lead to poor algorithmic performance.

This insight provides the mechanism by which incorporating monotonicity can lead to an increase in performance. This is because incorporating monotonicity rules out any demand curve with a region where demand increases as price increases. This allows for a large reduction in the space of possible demand curves, making the learning problem easier, especially when the optimal is a low price (that is, when many low reward, high prices need to be ruled out).

6. Analysis and Results

We now explain how the policies (algorithms) were implemented, specifically detailing the building blocks like consumer valuations, the number of arms, and the sequence of events in the multi-armed bandit. A summary of the benchmarks along with their characteristics is presented in Table 3.

Table 3 Overview of All Algorithms

Algorithm	Bayesian	Nonparametric	Dependence Across Arms	Implements Monotonicity
TS	✓			
N-TS	✓	✓		
UCB		✓		
GP-TS	✓	✓	✓	
GP-UCB	✓	✓	✓	
GP-TS-M	✓	✓	✓	✓
GP-UCB-M	✓	✓	✓	✓

Sequence of Events: We evaluate our proposed algorithm and benchmarks using a series of simulations based on a standard setup for evaluating bandit algorithms in pricing (Misra et al. 2019). Each simulation has the following structure. First, at each time period, a consumer with a WTP drawn from an unknown distribution arrives from a large pool of potential consumers with unit demand for the product. They are shown a price chosen by the algorithm (which has no specific knowledge about this consumer), and they decide to purchase if and only if their valuation for the product is greater than the price shown. The outcome (purchase or no purchase) is observed by the algorithm, which then updates its history of observations. We allow for the algorithm to update its price every 10 consumers.[24]

Varying the Number of Arms: In choosing the number of arms, the decision maker faces a trade-off between precision of the optimal chosen price, and the complexity (and time) of the learning problem. We evaluate the performance across different price sets normalized from 0 to 1 in equal intervals of 100 arms, 10 arms, and 5 arms.

Initializing the Algorithm: The algorithms are initialized with either a prior or some limited experimentation. For UCB, we assume a prior that encourages exploration by assuming that every untested price has been tested once and resulted in a purchase.[25] On the other hand, TS- and GP-variants can use uninformed priors so that a price can be chosen even without any data. However, to make the comparison more equivalent,[26] for GP-variants we have chosen to pick the first price randomly; once the first price has been chosen, the GP can be estimated from the data.

Valuation (WTP) Distributions: Right-skewed, Left-skewed and Symmetric: To evaluate the performance of various MAB policies, it is important to analyze different kinds of distributions of consumer valuations or WTP. We follow Misra et al. (2019) by analyzing distributions of various shapes using various parametrizations of the Beta distribution. Specifically, for a right-skewed distribution we use Beta(2,9), for a left-skewed distribution we use Beta(9,2), and for a symmetric distribution we use Beta(2,2). A full graphical description of the willingness to pay, the demand curve, and the profit curve for each simulation setting can be found in Figure 2. A monopolist who knows the demand curve would

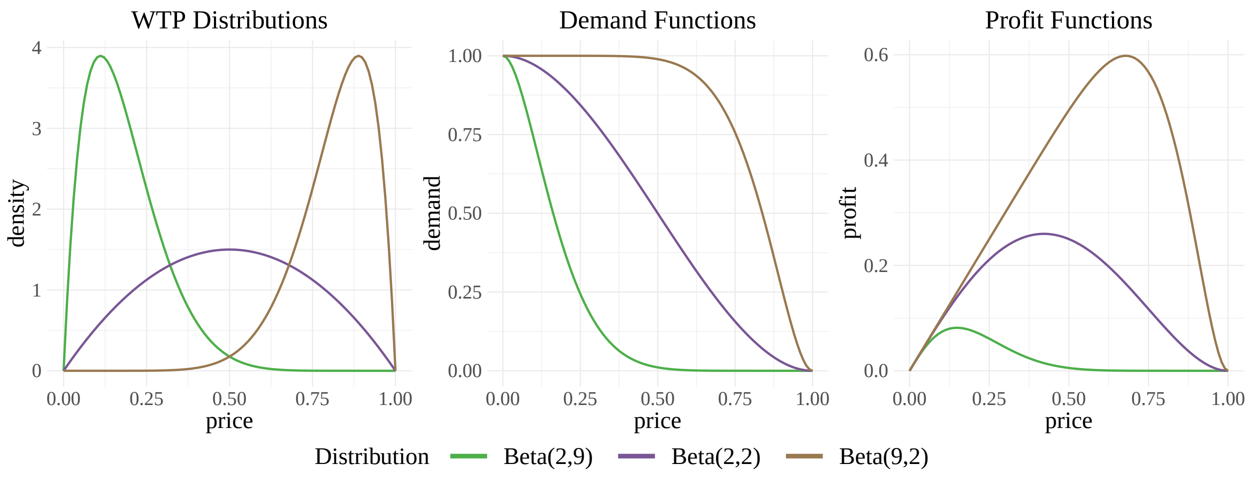
[24] While it is possible to change prices every period (i.e., for each consumer), in practice this may be too quick to employ. To better represent industry standards we change prices every 10 customers, as in Misra et al. (2019).

[25] The alternative is to test each price before using the UCB policy, however, our initialization priors allow for the possibility of not testing every price leading to an increase in algorithmic performance.

[26] There are slight differences in how monotonic and non-monotonic algorithms choose the first price from an uninformed prior.

set the optimal price to obtain the highest profit price. The true optimal prices range from 0.16 for Beta(2,9) to 0.42 for Beta(2,2) to 0.67 for Beta(9,2).

Figure 2 WTP Distributions and Demand and Profit Functions



6.1. Results

Our results present several variants among two main classes of algorithms (TS and UCB). Figure 3 illustrates the cumulative performance of each of the algorithms across time (or number of customers), while Table 4 gives the corresponding exact numbers from this figure at 500 and 2500 consumers. Table 5 details the uplift, or gains, from adding the two informational externalities relative to the baseline algorithms. Last, Figure 4 presents a histogram of prices played, providing insight into the behavior of the various algorithms.

The performances of the algorithms are shown in Figure 3, which reports the cumulative percent of optimal rewards, relative to the case when the optimal arm (price) is played.[27] In each subfigure, the horizontal black dotted line represents the maximum obtainable rewards given the price set (i.e., the ratio of the rewards from the best price within the price set and the true optimal reward given the underlying WTP distribution). Observe that this is not the same as the true optimal rewards, which is represented by 100%.[28] Visually, algorithms with no informational externalities are represented with dots, those with the first informational externality (i.e., the GP) are represented with dashes, and

[27] There are two sources of variation between simulations — one that depends on the algorithm itself, and another that depends on the exact WTP draws, from the true distribution. By using the expected rewards from playing a price, it lessens the variation caused by WTP draws allowing for a cleaner comparison of algorithmic performance. Misra et al. (2019) also use expected rewards in their analysis.

[28] As the number of arms grows, the optimal rewards given the price set will approach the true optimal rewards.

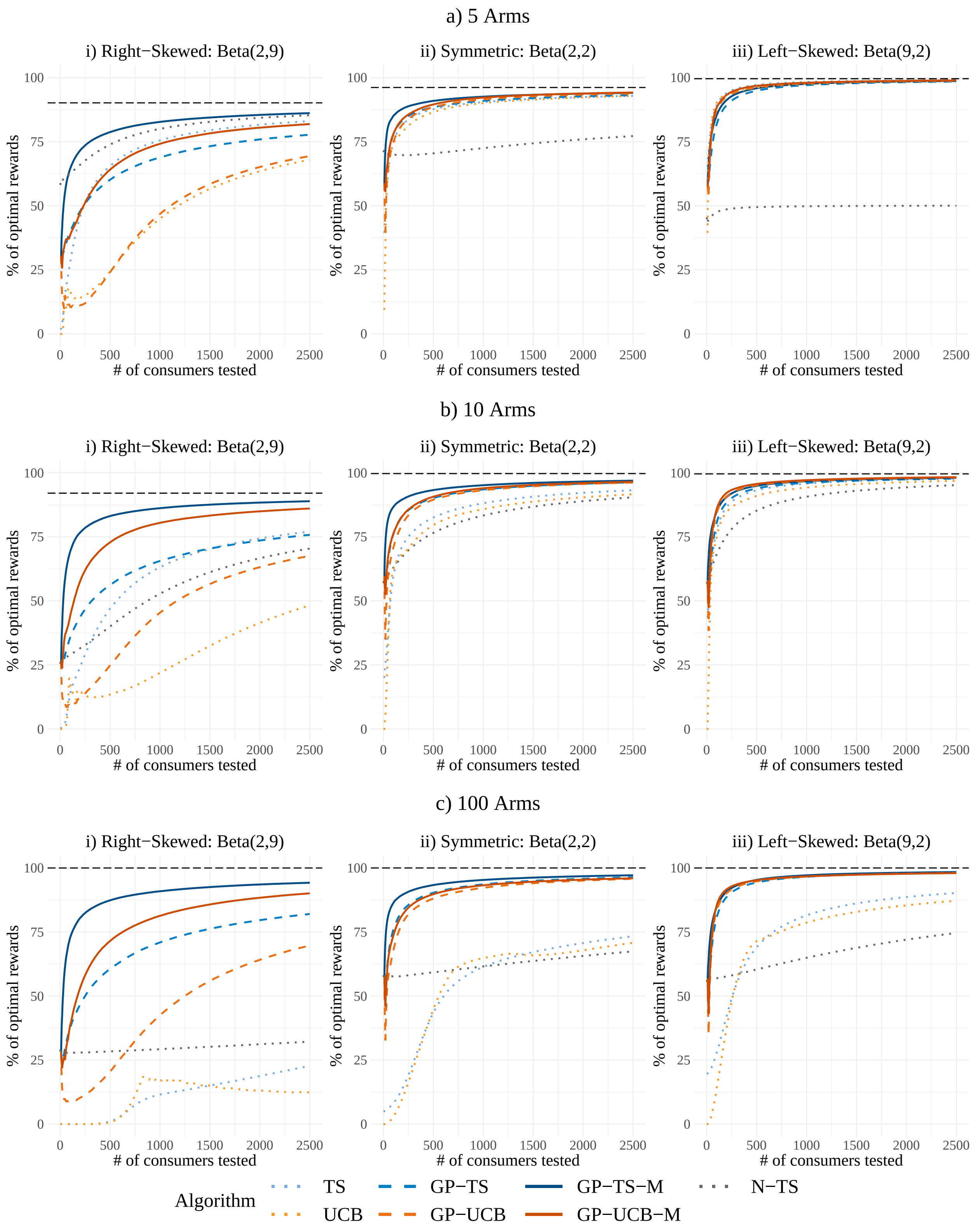
those with both informational externalities are represented with solid lines. UCB-variants are represented with warm (orange-based) colors, while TS-variants are represented with cool (blue-based) colors.

Table 4 Cumulative Percent of Optimal Rewards (Profits)

	After 500 Consumers							
	% of Price Set Optimal Reward				% of True Optimal Reward			
Algorithm	B(2,9)	B(2,2)	B(9,2)	Mean	B(2,9)	B(2,2)	B(9,2)	Mean
	5 Arms							
TS	72.7	91.2	97.6	**87.2**	65.6	87.7	97.2	**83.5**
GP-TS	66.7	92.0	95.3	**84.7**	60.2	88.4	94.9	**81.2**
GP-TS-M	87.3	94.6	96.3	**92.7**	78.7	90.9	95.9	**88.5**
N-TS	82.0	73.3	49.7	**68.3**	73.9	70.5	49.5	**64.6**
UCB	27.0	90.0	96.5	**71.1**	24.3	86.5	96.1	**69.0**
GP-UCB	26.6	92.0	96.6	**71.8**	24.0	88.5	96.2	**69.6**
GP-UCB-M	71.1	93.1	97.2	**87.1**	64.1	89.5	96.8	**83.5**
	10 Arms							
TS	51.0	82.8	93.6	**75.8**	46.9	82.5	93.2	**74.2**
GP-TS	61.1	90.4	94.3	**81.9**	56.2	90.1	93.9	**80.1**
GP-TS-M	90.3	93.5	95.3	**93.0**	83.1	93.3	94.9	**90.4**
N-TS	43.5	76.8	85.6	**68.6**	40.0	76.5	85.2	**67.3**
UCB	14.6	79.8	91.4	**61.9**	13.5	79.6	91.0	**61.3**
GP-UCB	27.0	89.8	95.6	**70.8**	24.8	89.5	95.2	**69.8**
GP-UCB-M	79.1	90.9	96.1	**88.7**	72.9	90.7	95.7	**86.4**
	100 Arms							
TS	1.0	43.5	69.0	**37.8**	1.0	43.5	69.0	**37.8**
GP-TS	60.6	90.3	94.3	**81.8**	60.6	90.3	94.3	**81.8**
GP-TS-M	87.4	93.3	95.3	**92.0**	87.4	93.3	95.3	**92.0**
N-TS	28.4	59.3	60.4	**49.3**	28.4	59.3	60.4	**49.3**
UCB	0.7	44.8	71.4	**39.0**	0.7	44.8	71.4	**39.0**
GP-UCB	20.5	88.1	94.8	**67.8**	20.5	88.1	94.8	**67.8**
GP-UCB-M	71.5	89.8	95.2	**85.5**	71.5	89.8	95.2	**85.5**
	After 2500 Consumers							
	% of Price Set Optimal Reward				% of True Optimal Reward			
Algorithm	B(2,9)	B(2,2)	B(9,2)	Mean	B(2,9)	B(2,2)	B(9,2)	Mean
	5 Arms							
TS	92.1	96.6	99.4	**96.1**	83.0	92.9	99.0	**91.7**
GP-TS	86.2	96.9	98.9	**94.0**	77.7	93.2	98.5	**89.8**
GP-TS-M	95.6	97.8	99.1	**97.5**	86.1	94.1	98.7	**93.0**
N-TS	94.6	80.3	50.2	**75.0**	85.3	77.2	50.0	**70.8**
UCB	75.4	96.6	99.1	**90.4**	68.0	92.9	98.7	**86.5**
GP-UCB	77.0	97.5	99.2	**91.2**	69.4	93.8	98.8	**87.3**
GP-UCB-M	90.8	98.0	99.3	**96.0**	81.9	94.2	98.9	**91.7**
	10 Arms							
TS	83.7	93.5	98.1	**91.8**	77.1	93.2	97.7	**89.3**
GP-TS	82.3	96.6	98.2	**92.4**	75.8	96.3	97.7	**89.9**
GP-TS-M	96.6	97.2	98.4	**97.4**	88.9	96.9	98.0	**94.6**
N-TS	76.5	90.7	95.6	**87.6**	70.4	90.4	95.2	**85.3**
UCB	52.4	91.9	97.3	**80.5**	48.3	91.6	96.9	**78.9**
GP-UCB	73.4	96.5	98.6	**89.5**	67.6	96.2	98.2	**87.3**
GP-UCB-M	93.5	96.8	98.7	**96.3**	86.1	96.5	98.3	**93.6**
	100 Arms							
TS	22.8	73.3	90.2	**62.1**	22.8	73.3	90.2	**62.1**
GP-TS	82.0	96.3	98.2	**92.2**	82.0	96.3	98.2	**92.2**
GP-TS-M	94.2	97.1	98.4	**96.6**	94.2	97.1	98.4	**96.6**
N-TS	32.2	67.4	74.6	**58.1**	32.2	67.4	74.6	**58.1**
UCB	12.4	70.8	87.2	**56.8**	12.4	70.8	87.2	**56.8**
GP-UCB	69.7	95.7	98.1	**87.8**	69.7	95.7	98.1	**87.8**
GP-UCB-M	90.1	96.0	98.0	**94.7**	90.1	96.0	98.0	**94.7**

Figure 3a) shows the results for five evenly spaced arms over $[0,1]$. Examining the left-most panel for the right-skewed Beta(2,9) distribution, we have a few noteworthy obser-

Figure 3 Cumulative Percent of Optimal Rewards (Profits)



Notes. The lines are the means of the cumulative expected percentage of optimal rewards across the 1000 simulations. The black horizontal line represents the maximum obtainable given the price set, while 100% represents the true optimal given the underlying distribution.

vations. First, while all algorithms eventually settle on the optimal price with sufficient learning, even at 2500 iterations there remains quite a substantial difference in performance across algorithms. Second, the UCB algorithm does not benefit quite as much from adding the first informational externality – incorporating a Gaussian process. However, we see that adding the second informational externality – monotonicity – adds significantly more benefit in terms of the cumulative rewards, improving the performance of the GP-UCB algorithm. Third, in the case of the TS-based algorithms, adding the first informational externality can actually make the rewards lower, i.e., GP-TS can be worse than just TS. It might seem counterintuitive that allowing the algorithm to use more information can reduce the rewards. However, the advantage of using a GP comes from learning from nearby arms, of which there is little opportunity from just five spaced out arms. However, one drawback of using a GP is that it shares a single noise hyperparameter, while the true underlying noise differs based on the price. If this drawback outweighs the benefit of sharing, GP-TS can actually perform worse. We address this problem by allowing for heteroscedastic noise in Appendix EC.10. Finally, we find that the nonparametric TS (N-TS) algorithm performs well relative to most other algorithms, but is worse than the GP-TS-M algorithm, which incorporates both informational externalities.

Next, we explore the symmetric and left-skewed distributions, i.e., Beta(2,2) and Beta(9,2) respectively. While we observe a similar ordering of algorithms as in the right-skewed Beta(2,9) case, we do not see such a substantial difference between the algorithms at 2500 iterations. There is one exception: unlike in the right-skewed case, nonparametric TS performs much worse in these distributions compared to the other algorithms, even baseline TS and UCB.

As we increase the number of arms from 5 to 10, a few interesting differences arise. First, with 10 arms, we observe that for all three distributions, the variation across algorithms has increased compared to the case with five arms, i.e., the worse performing algorithms now are much worse relative to the best performing ones. We observe that for all cases, that the rank ordering of the algorithms is relatively similar to the case with five arms. Notably, in the Beta(2,9) case, the first informational externality leads to GP-TS outperforming TS initially, before TS surpasses it narrowly by 2500 consumers. On the other hand, GP-UCB now greatly outperforms UCB. Similarly, in both the Beta(2,2) and Beta(9,2) case the difference between GP-TS (GP-UCB) and TS (UCB) increased from 5 to 10 arms. Comparatively, the advantage of the first informational externality increased as the number of arms increased from 5 to 10. Meanwhile, the second informational externality – monotonicity – continues to achieve a marked improvement over other algorithms in the Beta(2,9) case, while performing nearly identically to algorithms with the first informational externality in the Beta(2,2) and Beta(9,2) cases. Similar to the case with five arms, GP-TS-M is the best performing algorithm narrowly outperforming GP-UCB-M.

Moving from 10 to 100 arms, we observe the same patterns accentuated further. The divergence between the best performing (GP-TS-M) and worst performing algorithms (baseline) is substantial, with the best achieving 300% higher rewards than the worst. However, the rank ordering remains similar. N-TS performs either just above the baseline UCB and TS algorithms, or is the worst performer (in the left-skewed case).

Next, we examine the uplift quantitatively in Table 5. Uplift is the percentage improvement in cumulative rewards from including an informational externality. First, we examine the case with only five arms. We observe that for the UCB algorithm, the uplift from the first informational externality is positive but minimal, when averaged across the three valuation distributions. However, for the TS algorithm, the impact of incorporating the first informational externality is overall negative. As the number of arms increases to 10 and then 100, the uplift from including the first informational externality increases for all distributions, with the biggest gains being obtained in the right-skewed Beta(2,9) distribution, and the lowest gains for the left-skewed Beta(9,2) distribution. Adding the second informational externality, we observe that the greatest gains again accrue from the right-skewed distribution. However, in contrast to the first externality, adding monotonicity provides benefits that are consistent regardless of whether the number of arms is changed from 5 to 10 to 100. The above findings apply qualitatively when evaluated after 500 consumers (rounds), or 2500 consumers. However, impacts tend to be larger earlier on because the comparative gains from algorithms tend to happen in early rounds.[29] This also implies that the informational externalities are broadly useful no matter whether the manager has a low budget for experimentation (500 rounds) or a relatively high budget (2500 rounds), with higher relative gains in the former case.

[29] As eventually all algorithms will learn the optimal price.

Table 5 Uplift in Performance from Informational Externalities

		After 500 Consumers					
		TS			UCB		
		5 Arms	10 Arms	100 Arms	5 Arms	10 Arms	100 Arms
Uplift from 1st externality (GP compared to base algos)	**B(2,9)**	-8.0% (-8.6, -7.3)	20.6% (19.2, 22.1)	5940% (5840, 6040)	2.0% (0.0, 4.1)	92.1% (86.6, 97.6)	2720% (2650, 2790)
	B(2,2)	1.0% (0.6, 1.3)	9.4% (9.0, 9.8)	108% (107, 109)	2.4% (2.0, 2.7)	12.6% (12.1, 13.1)	96.6% (96.0, 97.2)
	B(9,2)	-2.4% (-2.5, -2.2)	0.8% (0.6, 1.0)	36.8% (36.6, 37.1)	0.1% (0.0, 0.3)	4.6% (4.5, 4.8)	32.7% (32.6, 32.8)
	Mean	**-2.8% (-3.1, -2.6)**	**8.2% (7.8, 8.5)**	**116% (115, 117)**	**0.9% (0.7, 1.2)**	**14.3% (13.9, 14.7)**	**73.9% (73.4, 74.4)**
Uplift from 2nd externality (GP-M compared to GP algos)	**B(2,9)**	31.5% (30.5, 32.5)	50.1% (48.4, 51.8)	45.5% (44.1, 47.0)	176% (170, 182)	237% (222, 252)	296% (281, 310)
	B(2,2)	2.9% (2.6, 3.2)	3.5% (3.2, 3.9)	3.4% (3.1, 3.8)	1.3% (1.0, 1.6)	1.4% (1.0, 1.8)	2.1% (1.6, 2.5)
	B(9,2)	1.0% (0.9, 1.2)	1.1% (1.0, 1.2)	1.1% (0.9, 1.2)	0.6% (0.5, 0.7)	0.5% (0.4, 0.6)	0.5% (0.4, 0.5)
	Mean	**9.6% (9.3, 9.8)**	**13.7% (13.3, 14.0)**	**12.6% (12.3, 12.9)**	**21.5% (21.1, 21.9)**	**25.6% (25.1, 26.0)**	**26.3% (25.9, 26.7)**
Uplift from both externalities (GP-M compared to base algos)	**B(2,9)**	20.4% (19.6, 21.3)	78.5% (76.9, 80.0)	8610% (8470, 8740)	174% (168, 179)	468% (457, 479)	975% (970, 980)
	B(2,2)	3.8% (3.5,4.1)	13.2% (12.7, 13.6)	115% (114,116)	3.6% (3.2, 3.9)	14.1% (13.6, 14.5)	100% (100, 101)
	B(9,2)	-1.4% (-1.5, -1.2)	1.9% (1.7, 2.1)	38.3% (38.0, 38.6)	0.8% (0.7, 0.9)	5.2% (5.0, 5.3)	33.3% (33.2, 33.4)
	Mean	**6.4% (6.1, 6.6)**	**22.9% (22.5, 23.2)**	**143% (143, 144)**	**22.6% (22.1, 23.0)**	**43.3% (42.9, 43.7)**	**119% (119, 120)**
		After 2500 Consumers					
		TS			**UCB**		
		5 Arms	10 Arms	100 Arms	5 Arms	10 Arms	100 Arms
Uplift from 1st externality (GP compared to base algos)	**B(2,9)**	-6.4% (-6.5, -6.2)	-1.6% (-1.9, -1.3)	262% (260, 264)	2.2% (1.9, 2.4)	40.4% (39.7, 41.1)	464% (462, 467)
	B(2,2)	0.3% (0.1, 0.4)	3.4% (3.2, 3.5)	31.4% (31.2, 31.6)	1.0% (0.9, 1.1)	5.0% (4.8, 5.1)	35.2% (35.0, 35.4)
	B(9,2)	-0.5% (-0.6, -0.5)	0.0% (0.0, 0.1)	8.8% (8.7, 8.9)	0.1% (0.0, 0.1)	1.3% (1.3, 1.4)	12.5% (12.4, 12.6)
	Mean	**-2.1% (-2.2, -2.0)**	**0.6% (0.5, 0.8)**	**48.4% (48.2, 48.6)**	**1.0% (0.9, 1.1)**	**11.1% (11.0, 11.3)**	**54.7% (54.5, 54.9)**
Uplift from 2nd externality (GP-M compared to GP algos)	**B(2,9)**	10.9% (10.6, 11.1)	17.4% (17.1, 17.8)	14.9% (14.6, 15.2)	18.0% (17.6, 18.5)	27.6% (27.1, 28.1)	29.4% (28.9, 30.0)
	B(2,2)	1.0% (0.9, 1.1)	0.7% (0.5, 0.8)	0.9% (0.7, 1.0)	0.4% (0.3, 0.5)	0.3% (0.2, 0.5)	0.3% (0.1, 0.4)
	B(9,2)	0.2% (0.2, 0.3)	0.3% (0.2, 0.3)	0.3% (0.2, 0.3)	0.1% (0.1, 0.2)	0.2% (0.1, 0.2)	-0.1% (-0.1, 0.0)
	Mean	**3.7% (3.6, 3.8)**	**5.5% (5.4, 5.6)**	**4.8% (4.7, 4.9)**	**5.3% (5.1, 5.4)**	**7.7% (7.5, 7.8)**	**7.8% (7.7, 7.9)**
Uplift from both externalities (GP-M compared to base algos)	**B(2,9)**	3.8% (3.6, 4.0)	15.4% (15.2, 15.6)	315% (313, 318)	20.5% (20.0, 21.0)	78.9% (78.1, 79.7)	629% (627, 632)
	B(2,2)	1.3% (1.1, 1.4)	4.0% (3.9, 4.2)	32.5% (32.3, 32.7)	1.4% (1.3, 1.6)	5.3% (5.1, 5.5)	35.6% (35.4, 35.8)
	B(9,2)	-0.3% (-0.4, -0.3)	0.3% (0.2, 0.4)	9.0% (8.9, 9.1)	0.2% (0.2, 0.3)	1.5% (1.4, 1.5)	12.4% (12.3, 12.5)
	Mean	**1.5% (1.4, 1.6)**	**6.2% (6.1, 6.3)**	**55.5% (55.4, 55.7)**	**6.3% (6.2, 6.4)**	**19.6% (19.5, 19.8)**	**66.8% (66.6, 66.9)**

Notes. The table provides mean uplifts (averaged across 1000 simulations) with their corresponding 99% confidence intervals calculated by using a paired t-test. Means are weighted.

Next, we evaluate the case when both informational externalities are added to determine whether they are substitutes, i.e., can one of them do the work or provide the value of the other in terms of increased rewards? We note that they lead to increases in different situations, and so the improvement from adding both greatly exceeds each individually.

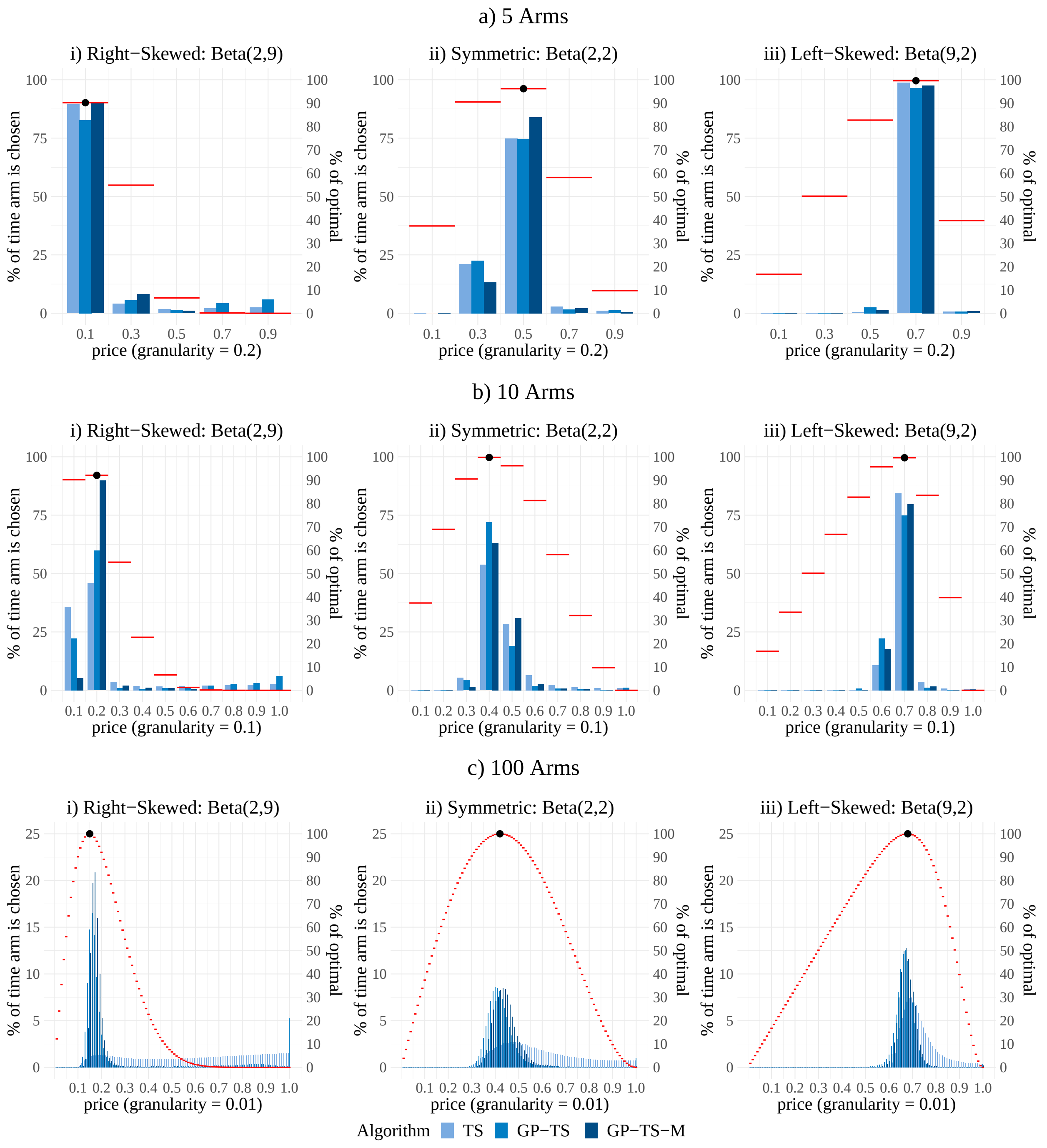
In summary, the results from cumulative rewards and uplift demonstrate a few important findings. First, we observe that the baseline TS and UCB algorithm do not perform very well across the variety of WTP distributions and the number of arms. Second, we find that the TS-based algorithms generally seem to perform better than the UCB-based algorithms. Third, adding the first informational externality (modeled using GP) is heavily impacted by the number of arms (giving an uplift across 48.4% across the three distributions) for 100 arms, but can make performance slightly worse (-2.1%) with just five arms. Fourth, adding the second informational externality (modeled with monotonicity) increases the rewards under all combinations of valuation distributions and number of arms. In contrast to the first informational externality, the uplift from monotonicity is consistent regardless of the number of arms. Fifth, the largest uplift from the combination of both informational externalities happened in the Beta(2,9) setting. As discussed in Section 5, this is because baseline algorithms perform poorly when the optimal price is low (Beta(2,9)), but excel when the optimal price is high (Beta(9,2)). Notably, even GP-TS-M performs slightly worse in the Beta(2,9) case compared to Beta(2,2) and Beta(9,2), but because the drop is much less than for other algorithms, the uplift is highest in this case. Overall, we find that whereas both informational externalities can contribute to greater rewards, the impact of adding monotonicity can be especially valuable since it is also less variable across conditions.

Mechanism: Investigating the Prices Chosen by Algorithms

In order to understand the mechanism by which each of the algorithms generates the rewards, we explore the set of prices (arms) played by each of the algorithms. In Figure 4, we have three sets of three panels. Across all of these graphs, there are both a left and right y-axis. The left y-axis represents the percentage of time each price was chosen, as shown by the blue bars. The right y-axis represents the percentage of the true optimal received by playing a price; the optimal price is marked with a solid black circle, and the red dashed lines (with values on the right axis) indicate the proportion of optimal rewards obtainable by choosing that price.

With regards to the first informational externality (including a GP), the biggest gains compared to the baseline algorithms happened when there were 100 arms, while for 5 arms the gains were negligible or even negative. The mechanism for this becomes clear by comparing the top row (5 arms) with the bottom row (100 arms). Since TS does not consider dependence across arms, it needs to learn each price individually. As the number

Figure 4 Histogram of Prices Played



Notes. Each subfigure has two y-axes with price as the common x-axis. The left y-axis is for the bar plots and represents the % of the time each arm is chosen (for 2500 consumers, with results averaged over 1000 different simulations). Meanwhile, the right y-axis corresponds to the right horizontal lines, which denote the % of the true optimal reward that would be obtained at the corresponding price. The black dot denotes which price is the optimal within the price set.

of prices increases, this makes the learning problem more difficult. On the other hand, GP-TS is able to pool across many low performing arms and move on to an area with more rewards more quickly. As the number of arms increases, these advantages can become quite large. On the other hand, with just 5 arms, TS is able to learn about the reward distribution at each price relatively quickly, so there is no advantage to using GP-TS.[30]

Meanwhile, for the second informational externality, the largest gains came in the Beta(2,9), case with positive uplifts becoming smaller as we move to the Beta(2,2) case and the Beta(9,2) case. In the top-left panel (5 arms, right-skewed), it can be seen that while all algorithms play the optimal price of 0.1 the majority of the time, both TS and GP-TS play prices 0.7 and 0.9, while GP-TS-M almost never plays such a price. As these arms provide nearly zero reward, they are quite costly to play, and this illustrates how GP-TS-M performs better.

To understand why this occurs, recall that since profits $= pD(p)$ even if the demand has the same uncertainty everywhere, the uncertainty for profits is higher at higher prices. This means that algorithms that do not consider monotonicity (like TS and GP-TS) will need to invest significant resources to properly learn that these high prices are not optimal. However, monotonicity rules out many cases, which TS and GP-TS cannot, as it is required that each demand curve be monotonically decreasing. This eliminates many possible curves from consideration in this noisy area that GP-TS cannot rule out. Impressively, this can make obtaining a sample demand curve with high purchase probabilities at high prices extremely rare, as it requires for the demand to be even higher at all lower prices. This means monotonicity can effectively remove the need to learn in high noise, low reward regions with very few data observations, showing the value of incorporating monotonicity.

This advantage, however, only applies when the optimal price is low. As we can see from left-skewed panels in Figure 4, other algorithms are able to quickly dismiss low reward, low price arms because the uncertainty is less for lower prices. Thus, as the optimal price becomes higher, the advantage from the second informational becomes smaller (though it continues to be positive across all simulations that we ran).

Robustness: We also conduct several analyses to determine the robustness of the method developed here. First, to check whether the performance improvements can be important in a practical case, in Appendix EC.7, we compare the algorithm performance using a demand curve estimated from field data. We observe that the results obtained using valuations from estimates using real data are quite comparable to the simulations, pointing to the practical value and applicability of the method. Since the GP-based method requires continuity of

[30] In fact, it may be negative, as discussed in Appendix EC.10.

the demand curve, we consider cases where the demand curve has discontinuities at known prices (left-digit bias), and find that monotonicity can help even in such a challenging situation (see Appendix EC.8). Finally, we observe that most bandit algorithms in the long-run tend to converge to the optimal rewards. To understand where the proposed bandit algorithms have a long-run or persistent advantage, we explore the case of time-varying (seasonal) demand in Appendix EC.9. We find that even in this case, informational externalities, especially monotonicity, can prove valuable for rewards over the long-run.

7. Conclusion and Future Research

We have proposed a method to achieve efficient and robust learning of the demand curve using reinforcement learning informed by microeconomic principles. In particular, we have proposed a new method that incorporates monotonicity into multi-armed bandits.

Our method is especially useful for managers who have limited time for experimentation. In particular, the benefits to our algorithm compared to baseline methods happen in early rounds (eventually all algorithms will learn the optimal). This reduction in needed experimentation time should make price experimentation more palatable to managers; less experiments means the potential for monetary loss is minimized and fewer customers are impacted. Additionally, our method allows for a greater set of prices to be tested. As a finer grid of prices can get closer to the true optimal, this could not only increase gains during the experiment, but also have a long-lasting profit impact if upon termination of the experiment a particular price is chosen for the long-run.

There are several avenues to further extend the applicability of this research. First, the method could be adapted to situations with either multiple units purchased or even multiple products, wherein consumer choice of one product could also have an informational externality in inferring the distribution of valuations for other related products. Second, if experimentation of price levels and demand responses across competing firms is available, this would be a naturally important setting. More broadly, the efficient learning of unknown demand curves with minimal impact of experimentation remains a widely shared goal across a number of markets, and we believe this present research contributes by closely connecting theory to develop such algorithms.

Funding and Competing Interests

All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript. The authors have no funding to report.

References

Aghion P, Bolton P, Harris C, Jullien B (1991) Optimal learning by experimentation. *The review of economic studies* 58(4):621–654.

Agrawal R (1995) Sample mean based index policies by o (log n) regret for the multi-armed bandit problem. *Advances in Applied Probability* 27(4):1054–1078.

Agrawal S, Goyal N (2012) Analysis of thompson sampling for the multi-armed bandit problem. *Conference on learning theory*, 39–1.

Ariely D (2010) Why businesses don't experiment. *Harvard business review* 88(4).

Auer P (2002) Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3(Nov):397–422.

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

Bastani H, Simchi-Levi D, Zhu R (2022) Meta dynamic pricing: Transfer learning across experiments. *Management Science* 68(3):1865–1881.

Bayati M, Hamidi N, Johari R, Khosravi K (2020) Unreasonable effectiveness of greedy algorithms in multi-armed bandit with many arms. *Advances in Neural Information Processing Systems* 33:1713–1723.

Bergemann D, Schlag KH (2008) Pricing without priors. *Journal of the European Economic Association* 6(2-3):560–569.

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

Botev Z, Belzile L (2021) *TruncatedNormal: Truncated Multivariate Normal and Student Distributions*. URL `https://CRAN.R-project.org/package=TruncatedNormal`, r package version 2.2.2.

Bubeck S, Munos R, Stoltz G, Szepesvári C (2011) X-armed bandits. *Journal of Machine Learning Research* 12(May):1655–1695.

Chapelle O, Li L (2011) An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 2249–2257.

Cheung WC, Simchi-Levi D, Wang H (2017) Dynamic pricing and demand learning with limited price experimentation. *Operations Research* 65(6):1722–1731.

Ching AT, Osborne M (2020) Identification and estimation of forward-looking behavior: The case of consumer stockpiling. *Marketing Science* 39(4):707–726.

Chou C, Kumar V (2024) Estimating demand for subscriptions: Identifying willingness to pay without price variation. *Marketing Science (Forthcoming)* .

Chowdhury SR, Gopalan A (2017) On kernelized multi-armed bandits. *International Conference on Machine Learning*, 844–853 (PMLR).

Cohen S, Treetanthiploet T (2021) Correlated bandits for dynamic pricing via the arc algorithm. *arXiv preprint arXiv:2102.04263* .

Cohen SN, Treetanthiploet T (2020) Asymptotic randomised control with applications to bandits. *arXiv preprint arXiv:2010.07252* .

Dann C, Mansour Y, Mohri M, Sekhari A, Sridharan K (2022) Guarantees for epsilon-greedy reinforcement learning with function approximation. *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 4666–4689 (PMLR).

Dubé JP, Misra S (2023) Personalized pricing and consumer welfare. *Journal of Political Economy* 131(1):131–189.

Duvenaud D (2014) *Automatic model construction with Gaussian processes*. Ph.D. thesis.

Erdem T, Keane MP (1996) Decision-making under uncertainty: Capturing dynamic brand choice processes in turbulent consumer goods markets. *Marketing science* 15(1):1–20.

Ferreira KJ, Simchi-Levi D, Wang H (2018) Online network revenue management using thompson sampling. *Operations research* 66(6):1586–1602.

Filippi S, Cappe O, Garivier A, Szepesvári C (2010) Parametric bandits: The generalized linear case. *Advances in neural information processing systems* 23.

Furman J, Simcoe T (2015) The economics of big data and differential pricing. *The White House President Barack Obama* .

Gittins J (1974) A dynamic allocation index for the sequential design of experiments. *Progress in statistics* 241–266.

Goldberg PW, Williams CK, Bishop CM (1997) Regression with input-dependent noise: A gaussian process treatment. *Advances in neural information processing systems* 10:493–499.

Gordon BR, Zettelmeyer F, Bhargava N, Chapsky D (2019) A comparison of approaches to advertising measurement: Evidence from big field experiments at facebook. *Marketing Science* 38(2):193–225.

Handel BR, Misra K (2015) Robust new product pricing. *Marketing Science* 34(6):864–881.

Hanssens DM, Pauwels KH (2016) Demonstrating the value of marketing. *Journal of marketing* 80(6):173–190.

Hauser JR, Urban GL, Liberali G, Braun M (2009) Website morphing. *Marketing Science* 28(2):202–223.

Hendel I, Nevo A (2006) Measuring the implications of sales and consumer inventory behavior. *Econometrica* 74(6):1637–1673.

Hill DN, Nassif H, Liu Y, Iyer A, Vishwanathan S (2017) An efficient bandit algorithm for realtime multivariate optimization. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1813–1821.

Hoban PR, Bucklin RE (2015) Effects of internet display advertising in the purchase funnel: Model-based insights from a randomized field experiment. *Journal of Marketing Research* 52(3):375–393.

Hoffman M, Brochu E, De Freitas N, et al. (2011) Portfolio allocation for bayesian optimization. *UAI*, 327–336 (Citeseer).

Huang J, Reiley D, Riabov N (2018) Measuring consumer sensitivity to audio advertising: A field experiment on pandora internet radio. *Available at SSRN 3166676* .

Huang Y, Ellickson PB, Lovett MJ (2022) Learning to set prices. *Journal of Marketing Research* 59(2):411–434.

Jindal P, Zhu T, Chintagunta P, Dhar S (2020) Marketing-mix response across retail formats: the role of shopping trip types. *Journal of Marketing* 84(2):114–132.

Kawale J, Bui HH, Kveton B, Tran-Thanh L, Chawla S (2015) Efficient thompson sampling for online matrix-factorization recommendation. *Advances in neural information processing systems*, 1297–1305.

Kersting K, Plagemann C, Pfaff P, Burgard W (2007) Most likely heteroscedastic gaussian process regression. *Proceedings of the 24th international conference on Machine learning*, 393–400.

Kleinberg R, Slivkins A, Upfal E (2008) Multi-armed bandits in metric spaces. *Proceedings of the fortieth annual ACM symposium on Theory of computing*, 681–690.

Lai TL, Robbins H (1985) Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics* 6(1):4–22.

Lambrecht A, Tucker C, Wiertz C (2018) Advertising to early trend propagators: Evidence from twitter. *Marketing Science* 37(2):177–199.

Li Z, Jamieson K, Jain L (2023) Optimal exploration is no harder than thompson sampling. *arXiv preprint arXiv:2310.06069* .

Liu X (2022) Dynamic coupon targeting using batch deep reinforcement learning: An application to livestream shopping. *Marketing Science* forthcoming.

Maatouk H, Bay X (2017) Gaussian process emulators for computer experiments with inequality constraints. *Mathematical Geosciences* 49(5):557–582.

Micchelli CA, Xu Y, Zhang H (2006) Universal kernels. *Journal of Machine Learning Research* 7(12).

Misra K, Schwartz EM, Abernethy J (2019) Dynamic online pricing with incomplete information using multiarmed bandit experiments. *Marketing Science* 38(2):226–252.

Murray I (2008) Introduction to gaussian processes. *Dept. Computer Science, University of Toronto* .

Nair H (2007) Intertemporal price discrimination with forward-looking consumers: Application to the us market for console video-games. *Quantitative Marketing and Economics* 5(3):239–292.

Oren SS, Smith SA, Wilson RB (1982) Nonlinear pricing in markets with interdependent demand. *Marketing Science* 1(3):287–313.

Osband I, Blundell C, Pritzel A, Van Roy B (2016) Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 4026–4034.

Pfeffer J, Sutton RI (2006) Evidence-based management. *Harvard business review* 84(1):62.

Rao RC, Bass FM (1985) Competition, strategy, and price dynamics: A theoretical and empirical investigation. *Journal of Marketing Research* 22(3):283–296.

Rebonato R, Jäckel P (2011) The most general methodology to create a valid correlation matrix for risk management and option pricing purposes. *Available at SSRN 1969689* .

Ringbeck D, Huchzermeier A (2019) Dynamic pricing and learning: An application of gaussian process regression. *Available at SSRN 3406293* .

Rothschild M (1974) A two-armed bandit theory of market pricing. *Journal of Economic Theory* 9(2):185–202.

Rubel O (2013) Stochastic competitive entries and dynamic pricing. *European Journal of Operational Research* 231(2):381–392.

Russo D (2016) Simple bayesian algorithms for best arm identification. *Conference on Learning Theory*, 1417–1418 (PMLR).

Russo D, Van Roy B (2014) Learning to optimize via posterior sampling. *Mathematics of Operations Research* 39(4):1221–1243.

Sahni NS, Nair HS (2020) Does advertising serve as a signal? evidence from a field experiment in mobile search. *The Review of Economic Studies* 87(3):1529–1564.

Schwartz EM, Bradlow ET, Fader PS (2017) Customer acquisition via display advertising using multi-armed bandit experiments. *Marketing Science* 36(4):500–522.

Shahriari B, Swersky K, Wang Z, Adams RP, De Freitas N (2015) Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE* 104(1):148–175.

Simester D, Hu Y, Brynjolfsson E, Anderson ET (2009) Dynamics of retail advertising: Evidence from a field experiment. *Economic Inquiry* 47(3):482–499.

Soysal GP, Krishnamurthi L (2012) Demand dynamics in the seasonal goods industry: An empirical analysis. *Marketing Science* 31(2):293–316.

Srinivas N, Krause A, Kakade SM, Seeger M (2009) Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995* .

Strulov-Shlain A (2019) More than a penny's worth: Left-digit bias and firm pricing. *Available at SSRN 3413019* .

Thompson WR (1933) On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25(3/4):285–294.

Tirole J (1988) *The theory of industrial organization* (MIT press).

Urteaga I, Wiggins CH (2022) Nonparametric gaussian mixture models for the multi-armed bandit. *arXiv preprint arXiv:1808.02932* .

Wang Z, Hu M (2014) Committed versus contingent pricing under competition. *Production and Operations Management* 23(11):1919–1936.

Williams CK, Rasmussen CE (2006) *Gaussian processes for machine learning*, volume 2 (MIT press Cambridge, MA).

Yu M, Debo L, Kapuscinski R (2016) Strategic waiting for consumer-generated quality information: Dynamic pricing of new experience goods. *Management Science* 62(2):410–435.

Zhang JZ, Netzer O, Ansari A (2014) Dynamic targeted pricing in B2B relationships. *Marketing Science* 33(3):317–337.

Zhang L, Chung DJ (2020) Price bargaining and competition in online platforms: An empirical analysis of the daily deal market. *Marketing Science* 39(4):687–706.

Electronic Companion Supplement

EC.1 Details of Benchmarks

EC.1.1 Upper Confidence Bounds

The *Upper Confidence Bound* (UCB) policy implementation in this paper matches that of Misra et al. (2019), which adapts UCB-tuned from Auer (2002) to pricing by scaling the exploration term by p_k. This is because the range of possible rewards differs (and is known) at each arm, which is not generally assumed.

$$\begin{aligned} p_k^{\text{UCB}} &= \arg\max_{p_k \in P} \left(\bar{\pi}_{kt} + p_k \sqrt{\frac{\log(t)}{n_{kt}} \min\left(\frac{1}{4}, V_{kt}\right)} \right) \\ V_{kt} &= \left(\frac{1}{n_{kt}} \sum_{\tau=1}^{n_{kt}} \pi_{k\tau}^2 \right) - \bar{\pi}_{kt}^2 + \sqrt{\frac{2\log t}{n_{kt}}} \end{aligned} \tag{EC.1}$$

EC.1.2 Thompson Sampling

Thompson sampling (*TS*) is a randomized Bayesian parametric approach. For each action, a reward distribution is specified a priori and updated by the history of past trials. In our setting, purchase decision is binary, which allows for the use of beta distributions (Chapelle and Li 2011, Agrawal and Goyal 2012). Then in each round, an action is picked according to the probability that it is optimal given the history of past trials. That is,

$$\text{Prob}(p_k|H_{t-1}) = \text{Prob}(E[\pi_{k,t}|p_k] > E[\pi_{k,t}|p_{k'}], \forall p_{k'} \neq p_k|H_{t-1}) \tag{EC.2}$$

The easiest implementation is to take a sample from each distribution each round and pick the one that gives the highest payoff. Once a price has been chosen, and purchase decision observed, the distribution $\text{Beta}(\alpha, \beta)$ is updated to be $\text{Beta}(\alpha+1, \beta)$ if a purchase was made and $\text{Beta}(\alpha, \beta+1)$ if no purchase was made. The algorithm is initialized by using $\text{Beta}(1,1)$ as an uninformative prior.

EC.2 Performance Metrics

The most common metric in the bandit literature is *regret*, which is defined as the difference between profits under full knowledge versus the expected profits from the policy in question (Lai and Robbins 1985). Formally, the cumulative regret of policy Ψ through time t is

$$\text{Regret}(\Psi, P, t) = \mathbb{E}\left[\sum_{\tau=1}^{t} (\pi^* - \pi_{k_\tau} | \Psi, P, H_{\tau-1}) \right] \tag{EC.3}$$

where P is the set of prices being considered, π^* is the ex-post maximum expected profit in a given round, and π_{k_τ} is the profit realized when price p_k is played in time period τ.

An alternative objective is to maximize the expected total reward (Gittins 1974, Cohen and Treetanthiploet 2020). Like regret, this is also an ex-post measure, as calculating the expected total reward requires knowledge of the true reward distribution. In comparison, for field experiments the true rewards distribution is not known, meaning only the observed total reward can be used as a comparison metric. Formally, in our simulations, the goal is to pick a decision rule, Ψ, that picks a sequence of prices from consideration set, P, that maximizes the total expected profit

$$\mathbb{E}\left[\sum_{\tau=1}^{t} \pi_{k_t} | P\right] \tag{EC.4}$$

EC.3 Overview of Gaussian Processes

This section provides an overview of Gaussian processes using general notation.[31] Consider a situation where the goal is to learn a function f at some test points X^* from some potentially noisy data $\mathcal{D} = \{X, \mathbf{y}\} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$ where each $\mathbf{x}_i \in \mathbb{R}^d$ and each $y_i \in \mathbb{R}$. The output data y_i gives a noisy signal of the true value of f at x_i; that is, $y_i = f_i + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma_y^2)$. For ease of exposition, we will set $\mu(\mathbf{x}) = 0$.[32] A prior probability distribution over the space of functions is set, which changes when training data is incorporated. For example, Figure EC.1 illustrates a situation with noisy data. As data is obtained, the space in which the true function could live becomes restricted. Accordingly, the range of uncertainty is smaller in areas closer to data points compared to those far away from data points (as shown by the shaded area which represents the 95% confidence intervals at the test points).

Formally, the assumption is that f is jointly Gaussian-distributed and completely defined by its mean, $\mu(\mathbf{x})$, and covariance function $k(\mathbf{x}, \mathbf{x}')$ such that $f \sim \text{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')$. The mean and covariance function are defined as follows (Williams and Rasmussen 2006):

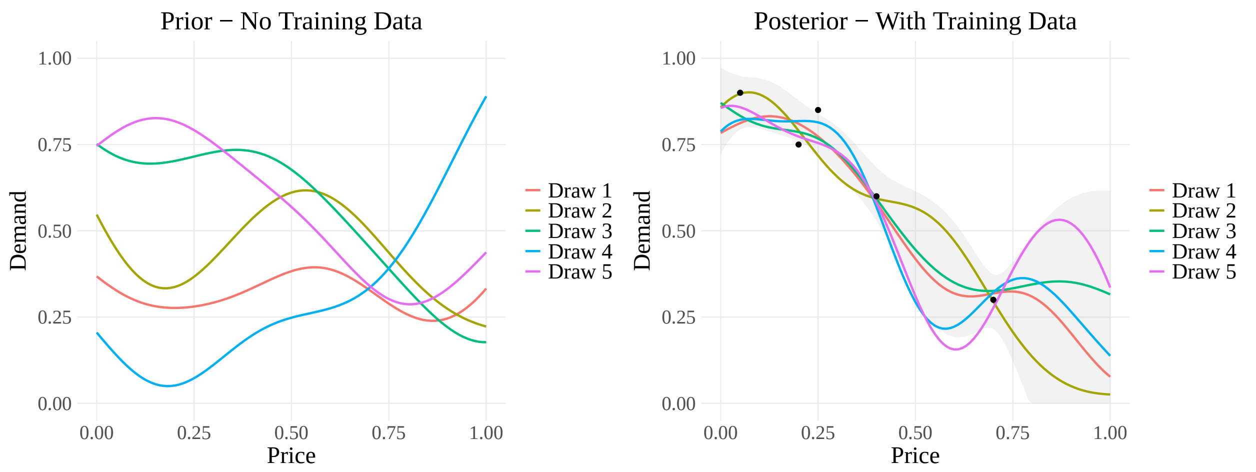
$$\mu(x) = \mathbb{E}[f(\mathbf{x})] \tag{EC.5}$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}') - \mu(\mathbf{x}')] \tag{EC.6}$$

[31] To adapt to our notation, X and $X*$ would become the price set P, and $\mathbf{f}^*$ would become D^*.

[32] If the prior mean function is non-zero, when $f \sim GP(\mu, k)$, the function $f' = f - \mu$ is a zero-mean Gaussian process $f' \sim GP(0, k)$. Hence, using observations from the values of f, one can subtract the prior mean function values to get observations of f', and do the inference on f'. Finally, after obtaining the posterior on $f'(X^*)$, one can simply add back the prior mean $\mu(X^*)$ to the posterior mean to obtain the posterior on f.

Figure EC.1 Random Samples from Gaussian Process With and Without Training Data



Notes. Lines represent five random draws from the GP in both the prior and the posterior. In the prior, the mean was set to 0.5. For both the prior and posterior, the RBF kernel was used with hyperparameters $l = 0.2, \sigma_f^2 = 0.08, \sigma_y^2 = 0.0016$. There were 101 test points $X^* = \{0, 0.01, 0.02, ..., 1\}$. The five draws from the posterior distribution are drawn from the GP with training data $X = \{0.05, 0.2, 0.25, 0.4, 0.7\}$, and $Y = \{0.9, 0.75, 0.85, 0.6, 0.3\}$. The shaded area provides the 95% confidence interval at each test point.

The kernel can be used to compute a covariance matrix $K(X^*, X^*)$ containing the covariance between all sets of test points, as well as a covariance matrix (either $K(X^*, X)$ or $K(X, X^*)$) between training and test cases. Then, the joint distribution of the training data X and the test points X^* can be written as follows (equation (2.21) in Williams and Rasmussen (2006)):

$$\begin{pmatrix} \mathbf{y} \\ \mathbf{f}^* \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \sigma_y^2 I & K(X,X^*) \\ K(X^*,X) & K(X^*,X^*) \end{bmatrix}\right) \tag{EC.7}$$

where $\mathbf{f}^* = f(X^*)$ is a random variable denoting the Gaussian process posterior prediction. It then follows from equations (2.22 - 2.24) in Williams and Rasmussen (2006) that

$$\mathbf{f}^* | X, \mathbf{y}, X^* \sim N(\mu(\mathbf{f}^*), \text{Cov}(\mathbf{f}^*)) \text{ where} \tag{EC.8}$$

$$\mu(\mathbf{f}^*) = K(X^*, X)[K(X,X) + \sigma_y^2 I]^{-1}\mathbf{y} \text{ and} \tag{EC.9}$$

$$\text{Cov}(\mathbf{f}^*) = K(X^*, X^*) - K(X^*, X)[K(X,X) + \sigma_y^2 I]^{-1} K(X, X^*) \tag{EC.10}$$

EC.4 Computational Issues

A computational issue that arises in fitting a posterior Gaussian process is that the matrix inversion is $\mathcal{O}(n^3)$, implying that it does not scale well to larger datasets. Typically, the training data would get larger with every purchase observation, but we are able to curtail this issue because input data points can only be observed at one of the test prices. This means we can estimate the GP by setting the input training data to be the set of test prices, and the associated output training data to be the sample purchase probabilities. That is, the noise parameter also needs to be adjusted as sample variance scales by the number of observations; it becomes $\boldsymbol{\sigma}_{\boldsymbol{y}}^2 = \{0.25/n_{1t}, ..., 0.25/n_{Kt}\}$. This approach means that the computational complexity will not increase as the number of purchase observations increases, but rather is dependent on the size of the initial set of test prices, P.

Additionally, one common issue when running GP algorithms is floating point precision errors, which can lead to negative eigenvalues, violating the positive semi-definite property of covariance matrices. We follow an approach devised by Rebonato and Jäckel (2011) to obtain the nearest covariance matrix.

EC.5 Implementation of Monotonic GP Bandits

For any continuous function $f:[0,1]\to\mathbb{R}$ the function $\tilde{f}(\cdot)\approx\sum_{j=0}^{N} f(u_j)h_j(\cdot)$ approximates f by linearly interpolating between the function values at the knots u_j. That is, each point of the estimate $\tilde{f}(\cdot)$ is just the weighted linear interpolation between two knots. For example, estimating $f(x)$ at $\frac{3}{8}$ gives $\tilde{f}(3/8)=(1/2)f(1/4)+(1/2)f(1/2)$.

EC.5.1 Proof of Proposition 1

LEMMA EC.1. *The distance between a continuous function D and its estimation via basis functions $D_N(p)=\sum_{j=0}^{N} D(\mu_j)h_j(p)$ converges to* 0 *as $N\to\infty$.*

From Lemma EC.1, a continuous demand D can be estimated via basis functions as $D(p)=\sum_{j=0}^{N} D(\mu_j)h_j(p)$. As, by assumption, the derivative of D is also continuous, then the same formula can be used to estimate D' as follows:

$$D'(p)\approx\sum_{j=0}^{N} D'(u_j)h_j(p) \tag{EC.11}$$

Additionally, by the Fundamental Theorem of Calculus

$$D(p)-D(0)=\int_0^p D'(t)dt \tag{EC.12}$$

Substituting (EC.11) into (EC.12) gives

$$D(p)\approx D(0)+\sum_{j=0}^{N} D'(u_j)\int_0^p h_j(t)dt \quad \square \tag{EC.13}$$

EC.5.2 Gaussian Process – Estimation of Derivatives and Intercept

The basis function method requires the estimation of the intercept and the derivatives at each of the prices in the consideration set. More formally, the goal is to estimate the posterior mean and covariance for $\{D(0), D'(p_1), ..., D'(p_k)\}$. Temporarily ignoring the intercept, the key is that the derivatives of a GP are also a GP. This means that D'^*, the posterior vector of derivatives of D^*, is

$$D'^* \sim N\left(\frac{d}{dp}\mu(D^*), \frac{d}{dp}\operatorname{Cov}(D^*)\right) \tag{EC.14}$$

Note as the values of D^* are only at our test points P, then the derivative only needs to be calculated with respect to P. This means that to estimate the posterior mean and covariance for $\{D(0), D'(p_1), ..., D'(p_k)\}$, the only necessary changes are to calculate the derivatives of the kernel function with respect to the test points.

We compute the partial derivatives of the kernel with respect to the prices as follows:

$$\frac{\partial k(p_i^*, p_j)}{\partial p_i^*} = \frac{\sigma_f^2}{l^2}(p_j - p_i^*)\exp\left(\frac{-(p_i^* - p_j)^2}{2l^2}\right) \tag{EC.15}$$

$$\frac{\partial k(p_i, p_j^*)}{\partial p_j^*} = \frac{\sigma_f^2}{l^2}(p_i - p_j^*)\exp\left(\frac{-(p_i - p_j^*)^2}{2l^2}\right) \tag{EC.16}$$

$$\frac{\partial^2 k(p_i^*, p_j^*)}{\partial p_i^* \partial p_j^*} = \frac{\sigma_f^2}{l^4}\left(l^2 - (p_i^* - p_j^*)^2\right)\exp\left(\frac{-(p_i^* - p_j^*)^2}{2l^2}\right) \tag{EC.17}$$

EC.6 Bounds for Joint Monotonic Algorithm

Consider a setting where $\mathcal{P} = \{p_1 \le p_2 \le \cdots \le p_d\}$ are a subset of prices that we choose as knots. We consider a Gaussian process for which draws are C^1 almost surely, and consider the joint distribution over draws f, f'. We define the set of monotonic functions,

$$M\{f \in C^1([0,1]) : f'(x) \le 0, x \in [0,1]\}.$$

Ideally we would like to restrict draws of our GP to M, but in general, since we can only evaluate our GP at a finite set of points, we instead insist that our function is monotonic at the set of knots,

$$M(\mathcal{P}) = \{f(\in C^1[0,1] : f'(p) \le 0, p \in \mathcal{P}\}$$

We denote the joint prior distribution over the function and the derivative $\Pi_0 = GP([0,0], K|)$, where K is the appropriate kernel.

Let $p^* = \arg\max_{p\in P} f(p)$ — note that since f is drawn from the underlying prior, p^* is a random variable. We define the Bayesian regret of our policy

$$BR_T = \sum_{t=1}^{T} \mathbb{E}[p^* f(p^*) - p_t f(p_t)]$$

where the expectation is over draws from the prior, reward noise, and any internal randomness of the algorithm.

Throughout the following, we refer to the truncated distribution as Π_t, and the untruncated distribution $GP([\mu_t, \mu_t'], K_t)$ as $\Pi_{t,u}$. Let the induced probability laws and expecation, with respect to the measure conditioned on the history $\mathcal{H}_t = \{(p_s, r_s)\}_{s=1}^t$, be $\mathbb{P}_t, \mathbb{E}_t$, and for the untruncated version, $\mathbb{P}_{t,u}, \mathbb{E}_{t,u}$. Critically we make the following assumption:

Assumption 1. The probability of returning a function monotonic on the knots is bounded below, i.e., there exists $c \geq 0$ such that

$$\mathbb{P}_{t,u}(f_t, f_t' \in) \geq c, \forall t \geq 1$$

Remark: Note that $P_{t,u}(f_t'(P) \geq 0)$ is equivalent to $\mathbb{P}_{D_t}(y \geq 0)$ for $x, y \sim N([\mu_t(P), \mu_t'(P)], K)$. This is an integral of a multivariate Gaussian over an open set. Since $f'(P) \geq 0$ by definition of the prior, if $\mu'(P) \to f'(P)$, we should expect this probability to actually increase with t. We discuss this further below.

THEOREM EC.1. *The Bayes Regret of Joint Algorithm is bounded by*

$$\begin{aligned} BR_t &\leq \mathbb{E}\left[\sqrt{\sum_{t=1}^{\infty} p_t^2}\right] \sqrt{\gamma_T \log(1+\sigma_0^{-2}) \log\left(\frac{T^2|A|}{\sqrt{2\pi}}\right)} + E_T + 1 \\ &\leq \sqrt{T\gamma_T \log(1+\sigma_0^{-2}) \log\left(\frac{T^2|A|}{\sqrt{2\pi}}\right)} + E_T + 1 \end{aligned}$$

where $E_T = \sum_{t=1} \mathbb{E}[\mu_t(p^*) - \mathbb{E}_t[f_t(p^*)]]$ *and* γ_T *is the mutual information of the Gaussian process (Srinivas et al. 2009).*

Interpreting the Regret: We remark that when $S = \mathbb{R}^d$, $\mathbb{E}_t[f(p^*)] = \mu_{t-1}(p^*)$ so E_T is 0. And so the final regret is of the form $O(\sqrt{\gamma_T T \log(T|P|)}$. Note that this regret is independent of the underlying constraint set.

To understand the impact of the underlying constraint set, we focus on the path-dependent regret term $\sum_{t=1}^{T} p_t^2$. In general, this quantity is less than the maximum price played times T, and is a tighter regret result compared to existing works.

We remark that the looseness in this result is primarily due to using loose tail bounds that do not effectively account for the constraint set. Future work could examine different and potentially tighter bounds.

Discussion of Assumption 1 and E_T. By Lemma EC.2,

$$\begin{aligned}\mu_t(p^*) - \mathbb{E}_t[f(p^*)] &\le \mu_t(p^*) - \mathbb{E}_{t,u}[f(p^*)\mathbf{1}\{f \in M\}] \\ &\le \mathbb{E}_{t,u}[f(p^*) - f(p^*)\mathbf{1}\{f \in M\}] \\ &\le \mathbb{E}_{t,u}[f(p^*)(1 - \mathbf{1}\{f \in M\})] \\ &\le \mathbb{E}_{t,u}[f(p^*)\mathbf{1}\{f \notin M\})]\end{aligned}$$

We define $\theta_0 = [f, f'] \sim \Pi_0$ to be the true parameters drawn from the prior, then existing results in the finite-dimensional bandit setting (Li et al. 2023, Russo 2016). We expect

$$\mathbb{P}_{t,u}(\mathbf{1}(f \notin M)) \approx e^{-t \min_{\theta \in M^c} \|\theta_0 - \theta\|^2_{K_t/t}}$$

In particular, if we can guarantee that $\|\theta_0 - \theta\|^2_{K_t/t}$ is bounded below (perhaps by sampling a uniform price $1/\sqrt{t}$ of the time), we see that the probability of not sampling a monotonic function decreases exponentially in time. This argument in particular implies that E_t should be finite.

Proof. We begin with the following lemma linking a distribution with its truncated version.

Lemma EC.2. *Let $p(x)$ be a distribution on $\mathbb{R}^d$. Let $S \subset \mathbb{R}^d$. Define the truncated pdf on $\mathbb{R}^d$, $p_S(x) = \mathbf{1}\{x \in S\}p(x)/\mu(S)$ where $\mu(S) = \int_{x \in S} p(x)$. Then given an event $E \subset \mathbb{R}^d$,*

$$\mathbb{P}_p(\mathbf{1}\{E \cap S\}) \le \mathbb{P}_{p_S}(E) \le \frac{1}{\mu(S)}\mathbb{P}_p(E)$$

and given a function $f(x) : \mathbb{R}^d \to \mathbb{R}$

$$\mathbb{E}_{p_S}(f(x)) \le \frac{1}{\mu(S)}\mathbb{E}_p(f(x))$$

T he lower bound is immediate since $\mu(S) \le 1$.

$$\begin{aligned}\mathbb{P}_{p_S}(E) &= \int_{x \in \mathbb{R}^d} \mathbf{1}\{x \in E\}\mathbf{1}\{x \in S\}p(x)/\mu(S) \\ &= \frac{1}{\mu(S)} \int_{x \in \mathbb{R}^d} \mathbf{1}\{x \in E\}\mathbf{1}\{x \in S\}p(x) \\ &\le \frac{1}{\mu(S)} \int_{x \in \mathbb{R}^d} \mathbf{1}\{x \in E\}p(x) \\ &\le \frac{1}{\mu(S)}\mathbb{P}_p(E)\end{aligned}$$

The result on expectations is almost immediate.

Define $U_t(p) := \mu_{t-1}(p) + \beta_{t-1}^{1/2}\sigma_{t-1}(p)$ where $\beta_t = \log(t^2 c^{-1}|P|/\sqrt{2\pi})$. Note that, conditioned on $\mathcal{H}_t$, the optimal action p^* and the action p_t selected by posterior sampling are identically distributed by Fact 5 (see below). In addition, U_t is deterministic conditioned on the history, so, $\mathbb{E}_t[U_t(p^*)] = \mathbb{E}_t[U_t(A_t)]$. Therefore,

$$\begin{aligned}\mathbb{E}[p^* f(p^*) - p_t f(p_t)] &= \mathbb{E}[\mathbb{E}_t[p^* f(p^*) - p_t f(p_t)]]\\ &= \mathbb{E}[\mathbb{E}_t[p_t U_t(p_t) - p^* U_t(p^*) + p^* f(p^*) - p_t f(p_t)]]\\ &= \mathbb{E}[\mathbb{E}_t[p_t U_t(p_t) - p_t f(p_t)] + \mathbb{E}_t[p^* f(p^*) - p^* U_t(p^*)]]\\ &= \mathbb{E}[p_t U_t(p_t) - p_t f(p_t)] + \mathbb{E}[p^* f(p^*) - p^* U_t(p^*)].\end{aligned}$$

Thus, we see that we can bound the Bayes-Regret as

$$BR(T) \le \sum_{t=1}^{T} \mathbb{E}[p_t U_t(p_t) - p_t f(p_t)] + \sum_{t=1}^{T} \mathbb{E}[p^* f(p^*) - p^* U_t(p^*)] \tag{EC.18}$$

$$\tag{EC.19}$$

We now focus on the first term,

$$\begin{aligned}p_t U_t(p_t) - p_t f(p_t) &= p_t U_t(p_t) - p_t \mu_t(p_t) + p_t \mu_t(p_t) - p_t f(p_t)\\ &= \mathbb{E}[p_t U_t(p_t) - p_t \mu_t(p_t)] + p_t \mu_t(p_t) - p_t f(p_t)\\ &\le p_t \beta_t^{1/2} \sigma_t(p_t) + p_t \mu_t(p_t) - p_t f(p_t)\\ &\le p_t \beta_t^{1/2} \sigma_t(p_t) + \mu_t(p_t) - f(p_t)\end{aligned}$$

Next,

$$\sum_{t=1}^{T} p_t \beta_t^{1/2} \sigma_t(p_t) \le \sqrt{\beta_T \sum_{t=1}^{\infty} p_t^2} \sqrt{\sum_{t=1}^{\infty} \sigma_t^2(p_t)} \tag{Cauchy-Schwartz}$$

A standard argument (see Srinivas et al. (2009)) shows that

$$\sum_{t=1}^{\infty} \sigma_t^2(p_t) \le \frac{\gamma_T}{\log(1+\sigma^{-2})}$$

Finally, we bound the second term of EC.18

$$\sum_{t=1}^{T} E[p^* f(p^*) - p^* U_t(p^*)] \le \sum_{t=1}^{\infty} \sum_{p \in P} \mathbb{E}_t[\mathbf{1}\{f(p) - U_t(p) \ge 0\}(f(p) - U_t(p))]$$

$$
\begin{aligned}
&\leq \sum_{t=1}^{\infty}\sum_{p\in P}\frac{1}{\mathbb{P}_{t,u}(M)}\mathbb{E}_{t,u}[\mathbf{1}\{f(p)-U_t(p)\geq 0\}(f(p)-U_t(p))]\\
&\leq \sum_{t=1}^{\infty}\sum_{p\in P}\frac{1}{c}\mathbb{E}_{t,u}[\mathbf{1}\{f(p)-U_t(p)\geq 0\}(f(p)-U_t(p))]
\end{aligned}
$$

Now, in the untruncated distribution, $\mathbb{E}_{t,u}[f(p)-U_t(p)] = -\beta_t^{1/2}\sigma_t^2(p)$, which is negative. Thus, using standard tail bounds Russo and Van Roy (2014),

$$
\begin{aligned}
\sum_{t=1}^{T} E[p^*f(p^*) - p^*U_t(p^*)] &\leq \frac{1}{c}\sum_{t=1}^{\infty}\frac{\sigma_t(p)}{\sqrt{2\pi}}e^{-\beta/2}\\
&\leq \frac{1}{c}\sum_{t=1}^{\infty}\frac{\sigma_t(p)}{t^2|P|c^{-1}} \leq 1
\end{aligned}
$$

The result follows from combining all the terms.

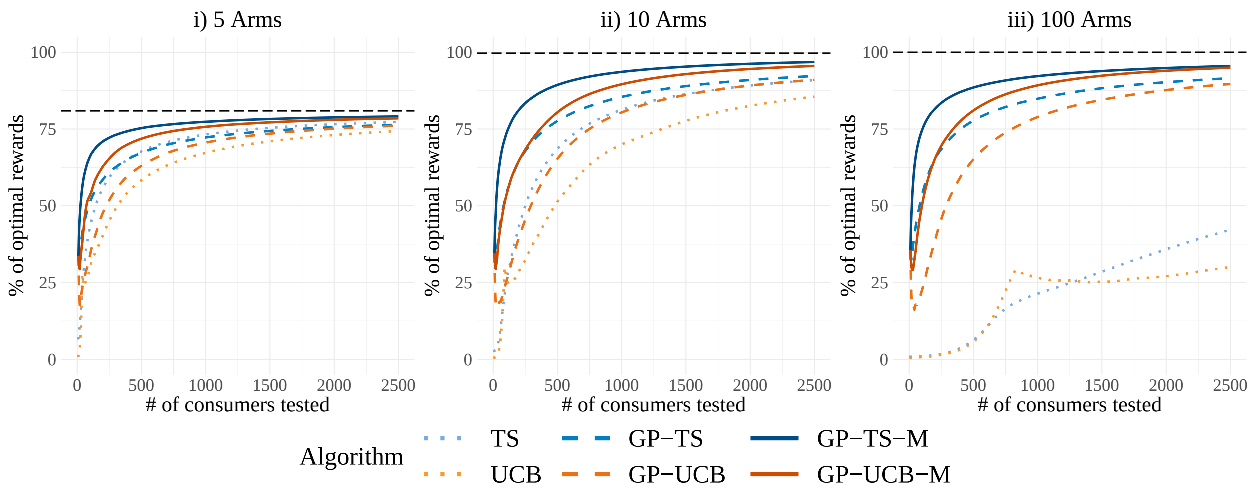
EC.7 Field Data

To further show the applicability and value of our methods when used with real world data, we tested out the algorithms on field data. The data comes from an empirical study of demand for a music streaming subscription service, where the distribution of WTP for a monthly plan is estimated (Chou and Kumar 2024; see Figure 2 on page 15 of that paper). We normalize the data by setting the price $p' = \frac{p}{1000}$ from their WTP distribution, which normalizes the prices to be in $[0,1]$.

From this WTP distribution, we are able to run the bandit algorithms using the same setup as in the simulations, but just by replacing the simulated WTP curve with the one derived empirically from field data. As the optimal price is relatively low price (0.21) within the set of prices tested, if the trend from the main results replicates, we would expect similar trends in uplifts from the informational externalities as in the Beta(2,9) case.

We find that the results are very consistent with those obtained for Beta(2,9) in the main simulations. The first informational externality (continuity implemented by GP) is negative for 5 arms, but positive for 10, and highly beneficial in the case of 100 arms. The second informational externality (implemented by specifying monotonicity) is positive for all the sets of arms, and is very consistent in terms of the improvement it offers. Overall, the results point to the validity and value of the method in practical applications.

Figure EC.2 Field Data: Cumulative Percent of Optimal Rewards



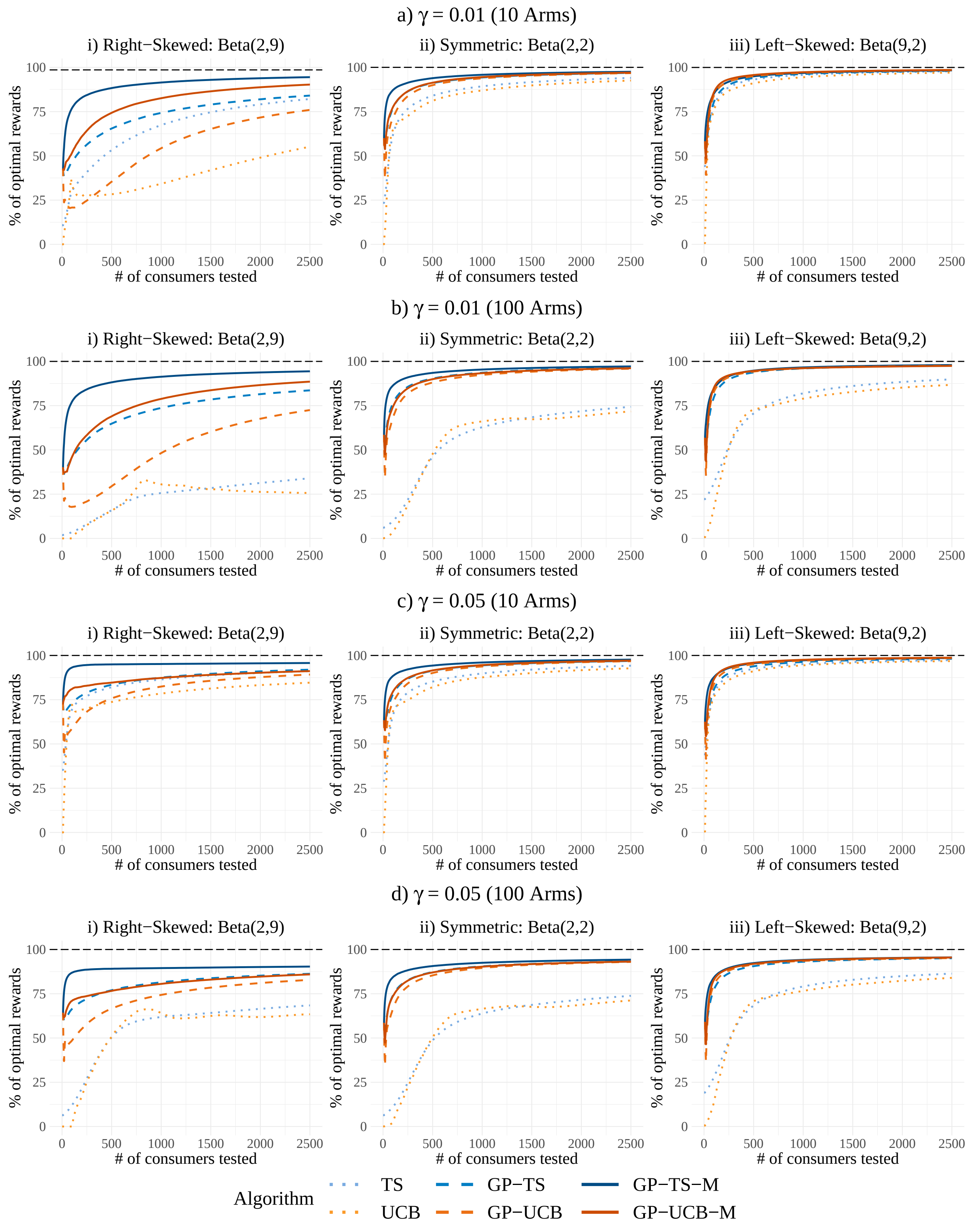
Notes. The lines are the means of the cumulative expected percentage of optimal rewards across the 1000 simulations. The black horizontal line represents the maximum obtainable given the price set, while 100% represents the true optimal given the underlying distribution. The true optimal price is 0.21.

EC.8 Discontinuous Demand: Left Digit Bias

We next discuss the case of the left digit bias. This is an interesting example where the demand function is not continuous and the points of discontinuity are typically at whole dollar values (Strulov-Shlain 2019). We model the demand by specifying discontinuities in the demand curve at known specific points in the $[0,1]$ continuum. Discontinuities occur between 0.09 and 0.10, and between 0.19 and 0.20, etc.

We evaluate how the algorithms perform when faced with learning this demand curve. The idea is that since the GP-based models rely on continuity and the idea of a continuous demand curve, the left-digit-bias case would serve as a stress test of the performance of the algorithms when the underlying assumptions are not satisfied. The discontinuity is parametrized by γ, where the discontinuity increases as γ increases. We observe that when we have 10 arms, the left digit discontinuity does not pose a substantial problem for the algorithm, because there is sufficient distance between the arms, relative to the degree of discontinuity.

Figure EC.3 Left Digit: Cumulative Percent of Optimal Rewards



Notes. The lines are the means of the cumulative expected percentage of optimal rewards across the 1000 simulations. The black horizontal line represents the maximum obtainable given the price set, while 100% represents the true optimal given the underlying distribution. γ is a measure of the size of the discontinuity spike that occurs at locations where the left-digit changes.

However, when we have 100 arms, we observe that the performance is negatively impacted, as seen in subfigure d)i) of Figure EC.3. Here, the cumulative percent of the optimal rewards curve flattens out well before the true optimal. The reason the algorithm obtains poor performance in this case is that it can get stuck choosing prices in a more stable area of decent reward returns, missing out on the true optimal, 0.39, which is followed by a precipitous drop at 0.40. However, because of the requirement of continuity, it causes the algorithm to mis-estimate the rewards at the optimal.

Nevertheless, after 2500 consumers, even with 100 arms and a high value of γ, GP-TS-M still greatly outperforms TS since the loss from the mis-estimate is less than the cost of having to learn each arm independently. As TS is not affected by continuity, with enough consumers it will eventually surpass the performance of GP-TS-M; however, it is unlikely any experiment would last this long in practice. In general, if it is suspected that there is a large left-digit discontinuity, we recommend using fewer arms, but at the discontinuities themselves[33] as there will be no mis-estimate issues, and large discontinuities tend to lead to the optimal price being one of those prices.

EC.9 Time Varying Demand

We now consider the case where there are demand changes depending on the season (Soysal and Krishnamurthi 2012). We model the demand curve as undergoing a shift after every X customers.[34] We denote the magnitude of the shift in the WTP distribution by δ, which results in a changed demand curve (shifted horizontally).

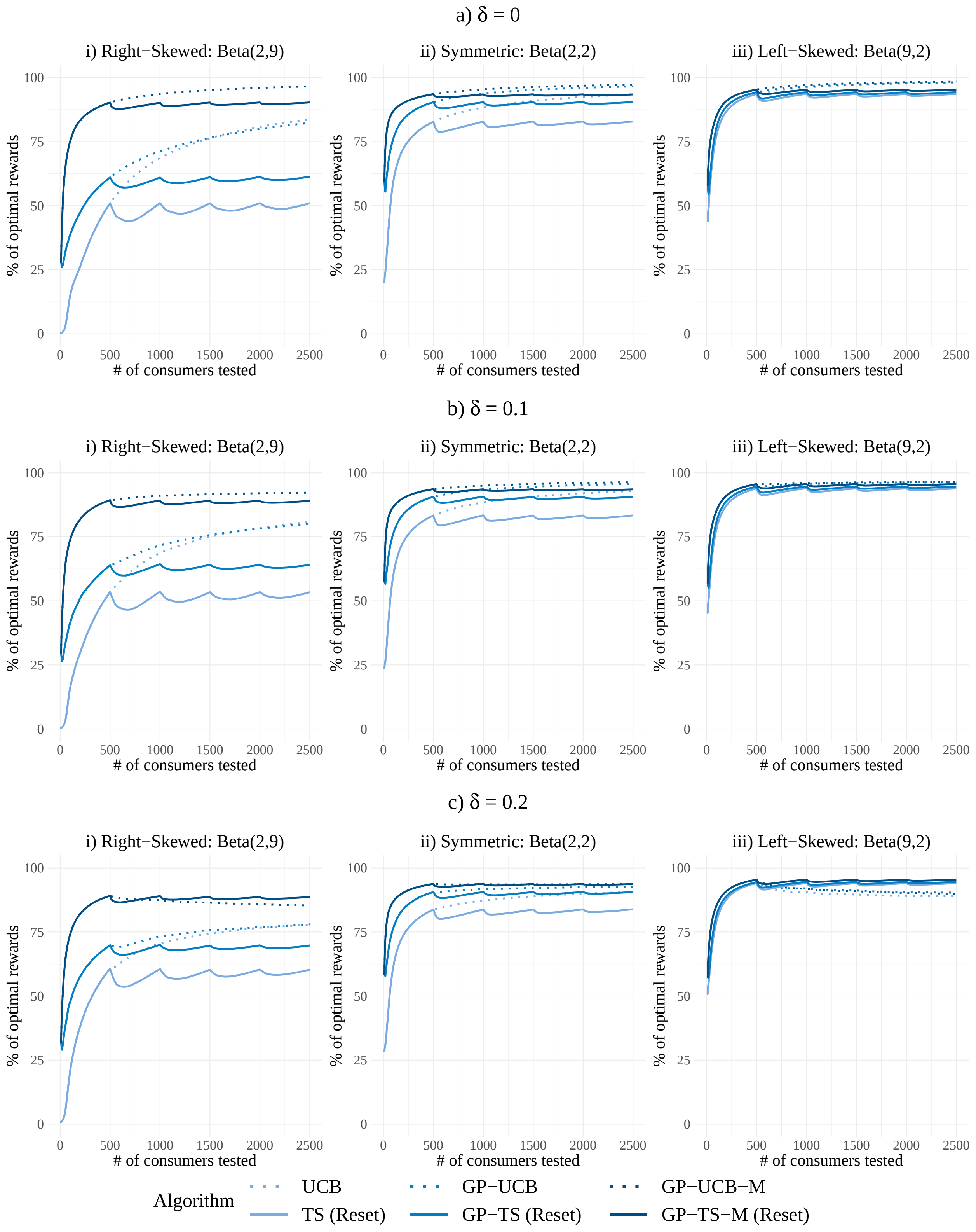
We explore the performance of the algorithms proposed in the paper, GP-TS and GP-TS-M, as well as their variants (GP-TS (Reset) and GP-TS-M (Reset)), which reset the learning process after every X consumers (seasonality cycle). The reset algorithm thus forgets all its history after every X periods, coinciding with shifts in the demand curve.

We observe that there are two separate effects that determine the effectiveness of the reset algorithm relative to the baseline ones. First, resetting is costly in terms of learning because the algorithm needs to re-learn the demand from the beginning (prior distribution). Second, on the flip side, resetting allows for more accurate learning when the underlying demand has changed. Thus, when δ is higher and the demand curve shifts are larger, it

[33] For example, a retailer may try 1.99, 2.99, etc.

[34] The model can be extended to having a distribution for the number of customers served, after which it undergoes a shift.

Figure EC.4 Time Varying: Cumulative Percent of Optimal Rewards (10 arms)



Notes. The lines are the means of the cumulative expected percentage of optimal rewards across the 1000 simulations. The black horizontal line represents the maximum obtainable given the price set, while 100% represents the true optimal given the underlying distribution. δ controls the size of the demand shock with a new time period starting every 500 consumers.

might be advantageous to reset, relative to GP-TS and GP-TS-M. The question of absolute and relative performance of the two sets of algorithms (with and without reset) depends on the interplay of the above effects.

The results from the case with time varying demand are detailed in Figure EC.4. The previously explained baseline algorithms without reset are illustrated using dotted lines, whereas the new reset algorithms are illustrated by solid lines. When there is no underlying shift in the demand curve ($\delta = 0$), then it is obvious that resetting would always be worse in terms of performance, which is confirmed in the top row of panels. We observe that resetting can provide higher performance as the underlying shift increases to $\delta = 0.1$ and $\delta = 0.2$. We also see that the rank ordering of the algorithms within the reset (or non-reset) category is stable, and does not change with either the valuation distribution or the number of arms.

Observe that the left-skewed Beta(9, 2) distribution is relatively easy to learn the optimal, as the optimal price is high. In this case, we find that resetting does not result in a large cost, and the algorithm with resetting can do better for every algorithm with enough of an underlying shift ($\delta = 0.2$).

However, when the learning problem is relatively more challenging, with the Beta(2,9) or right-skewed distribution, we observe that resetting can result in a very high cost. In particular, resetting with TS and GP-TS performs quite poorly compared to no reset even with high time varying shifts ($\delta = 0.2$). However, if the learning process is quick enough (GP-TS-M), the value of resetting and learning the true optimal can be greater than the cost of re-learning. Thus, the value of adding informational externalities (especially the monotonicity constraint) results in a persistent advantage.

EC.10 Heteroscedastic Noise

One reason that GP-based algorithms can perform poorly is that the uncertainty (noise) parameter is set to be homoscedastic when the true underlying noise is actually heteroscedastic. For example, the sample mean at a price where nearly every single customer purchases (or does not purchase) will be much less noisy than a price where customers are equally likely to purchase and not purchase. For UCB and TS, heteroscedastic noise is not much of a problem, as they learn each arm independently. However, GP-TS shares information across arms, and while this is beneficial (especially with a large number of nearby arms), this also means that the noise is shared across arms, which is particularly problematic when the noise is modeled to be homoscedastic. This explains why we observe in Table 5, in the case of five arms, that GP-TS performed worse than TS. When there is little to gain from sharing across arms (few arms that are far apart), the gain from sharing across arms is less than the negative caused by the misspecification of using homoscedastic noise. This appendix provides an alternative method of tuning the noise parameters, which allows for heteroscedasticity. Empirically, this change is a complement to monotonicity and led to a small increase in algorithmic performance in every simulation setting tested.

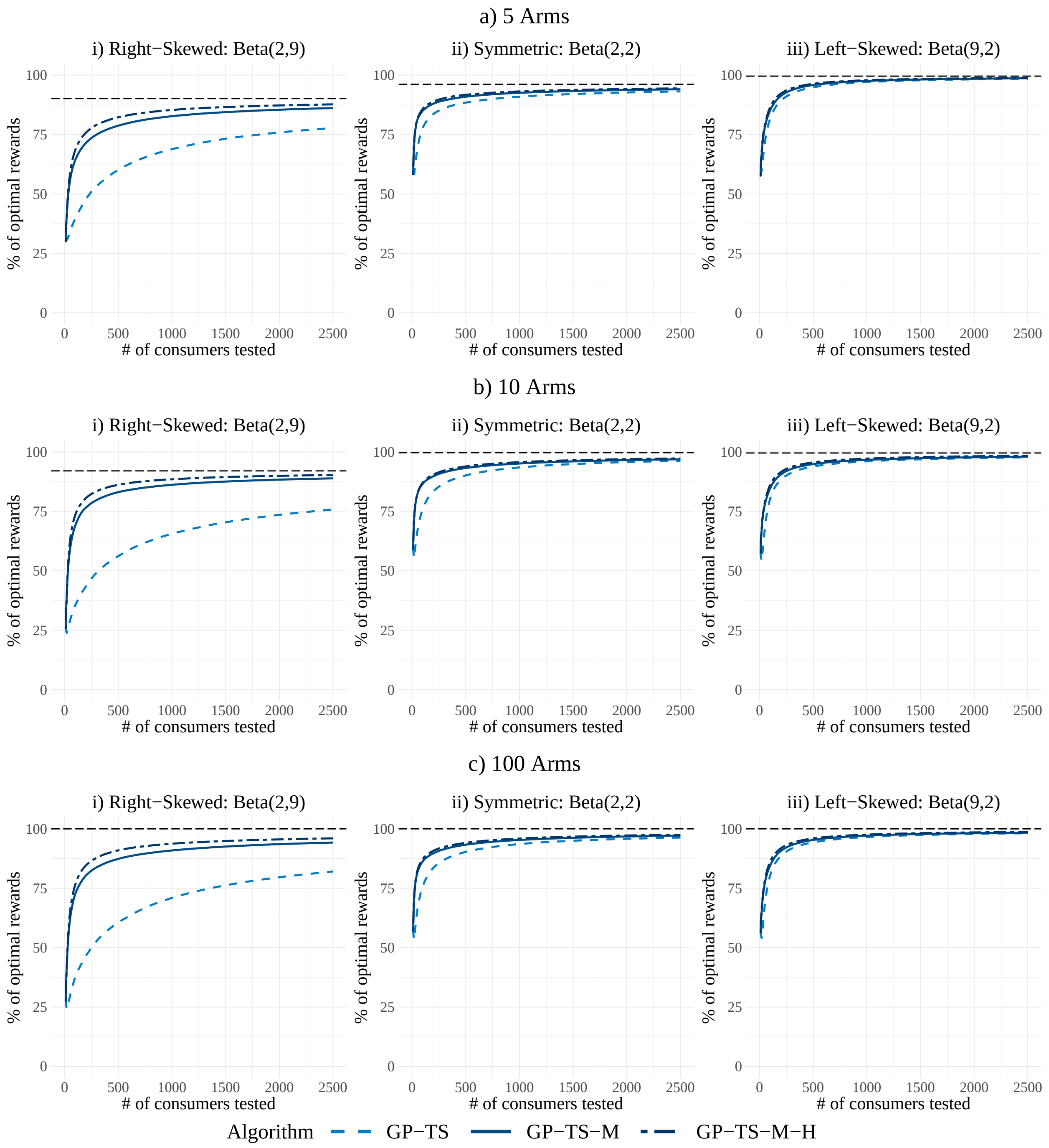
While there are approaches to estimate a GP with heteroscedastic noise (Goldberg et al. 1997, Kersting et al. 2007), it succumbs to the same issue as estimating the noise using MLE as our homoscedastic specification. That is, it can be difficult to identify the shape and noise hyperparameters (Murray 2008), which is quite problematic for bandits where an insufficient noise estimate can lead to the algorithm getting stuck on sub-optimal arms.

Another approach would be to model the error by specifying some underlying structural process. In our case, the noise of the data can be modeled depending on how likely a customer is to purchase. In general, we can write $\sigma_y^2 = g(D(p))$ for some unknown function g. However, noise depends on the underlying distribution which is completely unknown, making it unlikely that any suitable candidates for $g(\cdot)$ exist.

Implementation: Our estimation process for heteroscedastic noise works as follows. The first step is to estimate the GP using homoscedastic noise, which produces the results usually obtained. However, now we take a demand draw from this GP, which allows for the noise estimate to be calculated as the purchase decision is Bernoulli. A noise estimate can be calculated from the demand draw for each p in the test set as $\tilde{D}(p)(1 - \tilde{D}(p))$. Another GP is then derived using this estimation as the noise input (tuning the shape hyperparameters per usual). We call the implementation *GP-TS-H*, reflecting the flexible heteroscedastic specification. This implementation can easily be merged with *GP-TS-M* by using monotonic draws to create *GP-TS-M-H*. Importantly, this method provides a way to provide better estimates of the noise without the algorithm getting stuck underestimating for a particular price.

The results are presented in Figure EC.5. They show that including heterogeneity on top of monotonicity leads to a small increase across all simulations. Like for the other informational externalities, the effects are the largest for the Beta(2,9) case. This is because smaller noise hyperparameters at low reward, high prices synergize with monotonicity in reducing the space of potential demand curve samples.

Figure EC.5 Heteroscedasticity: Cumulative Percent of Optimal Rewards



Notes. The lines are the means of the cumulative expected percentage of optimal rewards across the 1000 simulations. The black horizontal line represents the maximum obtainable given the price set, while 100% represents the true optimal given the underlying distribution.