Operations Research

Assortment Optimization and Pricing Under the Multinomial Logit Model with Impatient Customers: Sequential Recommendation and Selection

Pin Gao, Yuhang Ma, Ningyuan Chen, Guillermo Gallego, Anran Li, Paat Rusmevichientong, Huseyin Topaloglu

CORRECTED VERSION OF RECORD; SEE LAST PAGE OF ARTICLE

informs
http://pubsonline.informs.org/journal/opre

OPERATIONS RESEARCH
Vol. 69, No. 5, September–October 2021, pp. 1509–1532
ISSN 0030-364X (print), ISSN 1526-5463 (online)

Crosscutting Areas

Assortment Optimization and Pricing Under the Multinomial Logit Model with Impatient Customers: Sequential Recommendation and Selection

Pin Gao,[a] Yuhang Ma,[b] Ningyuan Chen,[c] Guillermo Gallego,[d] Anran Li,[e] Paat Rusmevichientong,[f] Huseyin Topaloglu[b]

[a] School of Data Science, The Chinese University of Hong Kong, Shenzhen, China; [b] School of Operations Research and Information Engineering, Cornell Tech, New York, New York 10044; [c] Department of Management, University of Toronto Mississauga, Mississauga, Ontario L5L 1C6, Canada; [d] HKUST Industrial Engineering and Decision Analytics, Clear Water Bay, Hong Kong; [e] Department of Management, London School of Economics, London, WC2A 2AE, United Kingdom; [f] Marshall School of Business, University of Southern California, Los Angeles, California 90089

Contact: pgao@connect.ust.hk (PG); ym367@cornell.edu (YM); ningyuan.chen@utoronto.ca, https://orcid.org/0000-0002-3948-1011 (NC); ggallego@ust.hk, https://orcid.org/0000-0002-9664-3750 (GG); a.li26@lse.ac.uk, https://orcid.org/0000-0001-7001-2240 (AL); rusmevic@marshall.usc.edu, https://orcid.org/0000-0001-9584-4203 (PR); topaloglu@orie.cornell.edu, https://orcid.org/0000-0002-3049-6719 (HT)

Received: August 19, 2019
Revised: May 26, 2020; October 15, 2020
Accepted: January 21, 2021
Published Online in Articles in Advance: August 20, 2021

Subject Classifications: utility/preference: applications, choice functions; programming: integer: algorithms; marketing: choice models
Area of Review: Market Analytics and Revenue Management

https://doi.org/10.1287/opre.2021.2127

Abstract. We develop a variant of the multinomial logit model with impatient customers and study assortment optimization and pricing problems under this choice model. In our choice model, a customer incrementally views the assortment of available products in multiple stages. The patience level of a customer determines the maximum number of stages in which the customer is willing to view the assortments of products. In each stage, if the product with the largest utility provides larger utility than a minimum acceptable utility, which we refer to as the utility of the outside option, then the customer purchases that product right away. Otherwise, the customer views the assortment of products in the next stage as long as the customer's patience level allows the customer to do so. Under the assumption that the utilities have the Gumbel distribution and are independent, we give a closed-form expression for the choice probabilities. For the assortment-optimization problem, we develop a polynomial-time algorithm to find the revenue-maximizing sequence of assortments to offer. For the pricing problem, we show that, if the sequence of offered assortments is fixed, then we can solve a convex program to find the revenue-maximizing prices, with which the decision variables are the probabilities that a customer reaches different stages. We build on this result to give a 0.878-approximation algorithm when both the sequence of assortments and the prices are decision variables. We consider the assortment-optimization problem when each product occupies some space and there is a constraint on the total space consumption of the offered products. We give a fully polynomial-time approximation scheme for this constrained problem. We use a data set from Expedia to demonstrate that incorporating patience levels, as in our model, can improve purchase predictions. We also check the practical performance of our approximation schemes in terms of both the quality of solutions and the computation times.

Funding: G. Gallego was supported in part by the Research Grants Council, University Grants Committee, [Grant 16211619], and P. Rusmevichientong and H. Topaloglu were supported in part by the National Science Foundation Division of Civil, Mechanical and Manufacturing Innovation [Grants CMMI-1824860 and CMMI-1825406].
Supplemental Material: The online appendix is available at https://doi.org/10.1287/opre.2021.2127.

Keywords: choice model • multinomial logit • impatient customer • sequential recommendation

1. Introduction

A common assumption in traditional revenue-management models is that each customer enters the system with the intention to purchase a particular product. If this product is available for purchase, then the customer purchases it. Otherwise, the customer leaves the system without a purchase. In many settings, however, the customers observe the assortment of available products and choose and substitute within this assortment based on the features and prices of the offered products. In this case, the demand for a product depends on the availability of other products along with their features and prices. In revenue-management research, there has been a recent surge in using discrete-choice models to capture the fact that the customers choose and substitute among the available products. A large portion of these models work under the assumption that the customers view the entire assortment of products offered to them simultaneously, but it is clear that, in many cases, the customers incrementally view the assortment of offered products and make a purchase decision before viewing all the offered products. When purchasing products in online retail, for example, a customer may view the assortment of offered products on multiple web pages. When booking a healthcare appointment on the phone, the patient may be offered appointment slots gradually until the patient makes a choice. In both of these examples, the customer may make a purchase or leave without a purchase before viewing all the offered products or appointment slots. When customers view the assortment of offered products incrementally, the question is not only what assortment of products to offer, but also in which sequence to offer them.

We propose a variant of the multinomial logit model in which the customers incrementally view the assortment of offered products in multiple stages. We study assortment and pricing problems under this choice model. In our choice model, each customer has a different patience level sampled from a known distribution, which determines the maximum number of stages in which the customer is willing to view the assortment of products. In each stage, if the utility of a product in the current stage is larger than the utility of the outside option, then the customer purchases this product and leaves the system. Otherwise, the customer views the assortment of products in the next stage as long as the customer's patience level allows. Thus, in our model, customers impatiently leave for two reasons. First, a customer purchases a product in the current stage as soon as its utility exceeds a minimum acceptable utility even though there may be a product with a larger utility in a later stage. Second, a customer runs out of patience and leaves without viewing the entire assortment.

1.1. Main Contributions

Our main contributions are the formulation of the multinomial logit model with impatient customers as well as developing exact and approximate solution methods for assortment optimization and pricing problems under this choice model.

1.1.1. Multinomial Logit Model with Impatient Customers. We propose a new variant of the multinomial logit model with impatient customers. The choice model is based on random utilities. A customer arriving to the system associates random utilities with the products. Furthermore, the customer has a minimum acceptable utility and a patience level, which are also both random. We refer to the minimum acceptable utility as the utility of the outside option. The utilities of the products and the outside option are independent and have the Gumbel distribution with the same scale parameter. The patience level of the customer has a general distribution over the support $\{1, \dots, m\}$ for a fixed integer m and is independent of the utilities. The customer incrementally views the assortment of offered products in multiple stages. In each stage, if the product with the largest utility provides larger utility than the outside option, then the customer purchases this product, and the choice process terminates. If the utilities of all products that the customer views before running out of patience are smaller than the utility of the outside option, then the customer leaves without a purchase. Because the patience level of a customer is at most m, we choose pairwise disjoint assortments that we offer over m stages. We give a closed-form expression for the choice probability of each product under any assortment (Theorem 2.1).

1.1.2. Assortment Optimization. In the assortment-optimization problem, each product has a fixed revenue, and the goal is to find a revenue-maximizing sequence of assortments to offer. For this problem, we give a polynomial-time algorithm using the following steps. First, we show that there exists a revenue-ordered optimal solution. That is, letting n be the number of products and r_i be the revenue of product i, indexing the products so that $r_1 \geq r_2 \geq \dots \geq r_n$, the optimal assortment to offer in stage k is of the form $\{j_k^* + 1, \dots, j_{k+1}^*\}$ for some $0 = j_1^* \leq j_2^* \leq \dots \leq j_m^* \leq j_{m+1}^*$ (Theorem 3.1). Second, exploiting the revenue-ordered property, we give a dynamic program that finds the best sequence of revenue-ordered assortments using $O(mn^2)$ operations (Theorem 3.2).

1.1.3. Joint Pricing and Assortment Optimization. In the pricing setting, the mean utility of a product depends on its price. Following the standard assumption in the pricing literature, we assume that the products have the same price sensitivity; see Song and Xue (2007) and Li and Huh (2011). We start with the case in which the sequence of offered assortments is fixed and the goal is to find the revenue-maximizing prices. The expected revenue is not concave in the prices, but we give a reformulation in which the decision variables are the probabilities that a customer reaches different stages. We show that the expected revenue is concave in these decision variables, and we can recover the optimal prices after solving our reformulation (Theorem 4.2).

Next, we consider the case in which both the sequence of offered assortments and the prices are decision variables. We give an approximation algorithm that obtains at least 87.8% of the optimal expected revenue (Theorem 4.4). This approximation algorithm is based on showing that, if we offer all products in the first stage and compute the corresponding optimal prices, then the solution that we obtain has an 87.8% performance guarantee. In our computational experiments, starting from such a solution that is obtained by offering all products in the first stage, we use a neighborhood search algorithm to further improve the quality of this solution.

1.1.4. Space Constraints. We consider the assortment-optimization problem in which each product occupies a certain amount of space and there is a constraint on the total space consumption of the offered products. We give a fully polynomial-time approximation scheme (FPTAS) (Theorem 5.1). In the special case in which there is a constraint on the total number of offered products, we can improve the running time of our FPTAS. We also give an exact algorithm whose running time depends exponentially on the number of stages m and polynomially on the number of products n. A constraint on the total space consumption or the total number of offered products may arise, for example, when we want to avoid overwhelming a patient with too many appointment slot options or when we have a limited budget and offering a product requires a capital investment.

1.1.4.1. Numerical Results. Using a data set from Expedia, we check the performance of our choice model to predict customer purchases when compared against the standard multinomial logit benchmark. We use two metrics. The first metric is out-of-sample log-likelihood. The second metric is the fraction of customers whose bookings are predicted correctly. In the first and second metrics, our choice model improves upon the benchmark by, respectively, 1.95% and 4.49%, on average. Also, we test the practical performance of our approximation schemes for joint pricing and assortment optimization as well as for assortment optimization under space constraints.

In many online retail settings, the products are offered on multiple web pages, but the number of products on a web page is at the discretion of the retailer because the products are simply presented as a list as on Amazon, for example. Our unconstrained and constrained assortment-optimization problems as well as our joint pricing and assortment-optimization problem find applications in such settings. Our choice model is motivated by the satisficing behavior of customers, especially when purchasing leisure products, such as hotel rooms, when the customer directly proceeds to purchasing a product once the utility of the product exceeds a minimum acceptable utility.

1.2. Literature Review

There is recent assortment optimization work in which customers view only a portion of the offered assortment because of either search behavior or consideration sets. In Gallego et al. (2021), the customers decide on the number of web pages to view based on an exogenous distribution and choose within the entire assortment on these web pages according to a general choice model. Wang and Sahin (2018) consider a model in which the customers focus on a portion of the products by trading off the expected utility from the purchase with the search effort, but they do not view the assortment incrementally. Derakhshan et al. (2018) examine a product ranking problem in which the customers build a consideration set as a function of the search cost. In Aouad and Segev (2018), each customer views a random number of web pages and makes a choice within these web pages according to the multinomial logit model. The customers do not view the products sequentially. Aouad et al. (2019) focus on a setting in which each product is included in the consideration set of a customer with a fixed probability. In Feldman et al. (2019), the choice model is based on short preference lists, corresponding to the case with small consideration sets. In all these papers, the assortment-optimization problems are NP-hard, and the authors give approximation methods.

Liu et al. (2021) and Feldman and Segev (2019) study assortment-optimization problems under another variant of the multinomial logit model with multiple stages. The two papers use the same choice model in which the utility of the outside option is resampled when the customer considers the products at each stage during the course of the customer's choice process, and the utilities of the outside option at different stages are independent. Thus, a customer may associate a high utility with the outside option in one stage and a low utility in another stage. Under this choice model, it is NP-hard to find the revenue-maximizing sequence of assortments to offer. Both papers give approximation schemes, the main difference being that the running time of the approximation scheme depends exponentially on the number of stages m in Liu et al. (2021) and polynomially in Feldman and Segev (2019). In our choice model, the utility of the outside option is sampled once at the beginning of the choice process of the customer. We give a polynomial-time exact algorithm.

Gallego et al. (2004) and Talluri and van Ryzin (2004) study the assortment-optimization problem under the standard multinomial logit model and show that it is optimal to offer a revenue-ordered assortment. Rusmevichientong et al. (2010), Wang (2012), Jagabathula (2016), and Sumida et al. (2019) impose various constraints on the offered assortment. Bront et al. (2009), Mendez-Diaz et al. (2014), and Rusmevichientong et al. (2014) consider the problem under a mixture of multinomial logit models. Flores et al. (2019) use a two-stage multinomial logit model in which the products that can be offered in each of the two stages are fixed a priori. For the pricing problem, Song and Xue (2007), Hopp and Xu (2005), and Li and Huh (2011) show that the expected revenue is concave in the product market shares and the optimal prices of the products exceed their marginal costs by the same markup as long as the products have the same price sensitivity. Zhang et al. (2018) show that these two results hold under all generalized extreme value models.

We limit our literature review to the multinomial logit model, but we refer to Farias et al. (2013), Davis et al. (2014), Gallego and Wang (2014), Aouad et al. (2016), Blanchet et al. (2016), Desir et al. (2016b), and Li and Webster (2017) for work under other choice models.

1.3. Organization

In Section 2, we define our choice model and derive an expression for the choice probabilities. In Section 3, we consider the unconstrained assortment-optimization problem. In Section 4, we examine the joint pricing and assortment problem. In Section 5, we study constraints on the space consumption of the offered products. In Section 6, we give computational experiments.

2. Multinomial Logit Model with Impatient Customers

We describe our choice model and give an expression for the choice probabilities of the products. The set of products is $\mathcal{N} = \{1, \ldots, n\}$. The set of stages is $\mathcal{M} = \{1, \ldots, m\}$. We use $(S_1, \ldots, S_m)$ to denote the sequence of assortments that we offer over all m stages, in which $S_k \subseteq \mathcal{N}$ is the assortment that we offer in stage k. The assortments that we offer in different stages are disjoint, so $S_k \cap S_\ell = \emptyset$ for all $k \neq \ell$. The utility of product i is given by the random variable U_i, which has the Gumbel distribution with location-scale parameters $(\mu_i, 1)$. Letting $v_i = e^{\mu_i}$, we refer to v_i as the preference weight of product i. The utility of the outside option is given by the random variable U_0, which has the Gumbel distribution with location-scale parameters $(0, 1)$. The patience level of a customer is given by the random variable Y taking values in $\mathcal{M}$. A customer with patience level k is willing to view the assortments in the first k stages. We let $\lambda_k = \mathbb{P}\{Y \geq k\}$, so $1 = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m > 0$. The random variables $\{U_i : i \in \mathcal{N}\}$, U_0, and Y are independent. In our choice model, an arriving customer is characterized by the utilities the customer associates with the different products and the outside option along with the customer's patience level, all sampled from their distributions.

As stated in the introduction, the utility of the outside option corresponds to the minimum acceptable utility for the customer. A customer chooses among the products by sequentially viewing the assortments in each stage. Given that the customer is currently in stage k, if the product with the largest utility in stage k provides larger utility than the outside option, then the customer purchases this product. Otherwise, the customer moves on to stage $k+1$. If stage $k+1$ is beyond the patience level of the customer, then the customer leaves without a purchase. Otherwise, the customer views the products in stage $k+1$. The customer leaves the system for two reasons. First, if the product with the largest utility in stage k provides larger utility than the outside option, then the customer purchases this product right away even though there may be a product in a subsequent stage with larger utility. Second, because of the customer's patience level, a customer may not view all assortments in all stages. As a function of the assortments $(S_1, \ldots, S_m)$, let $\phi_i^k(S_1, \ldots, S_m)$ be the probability that a customer chooses product $i \in S_k$. In the next theorem, we give an expression for this choice probability. Throughout the paper, we let $V(S) = \sum_{i \in S} v_i$.

Theorem 2.1 (Choice Probabilities). *If we offer assortments $(S_1, \ldots, S_m)$ over m stages, then a customer purchases product $i \in S_k$ with probability*

$$\phi_i^k(S_1, \ldots, S_m) = \frac{\lambda_k v_i}{\left(1 + \sum_{\ell=1}^{k-1} V(S_\ell)\right)\left(1 + \sum_{\ell=1}^{k} V(S_\ell)\right)}.$$

Proof. Letting X_1 and X_2 be independent Gumbel random variables with location-scale parameters $(\mu_1, 1)$ and $(\mu_2, 1)$, we use three properties of Gumbel random variables. First, $\max\{X_1, X_2\}$ is a Gumbel random variable with location-scale parameters $(\log(e^{\mu_1} + e^{\mu_2}), 1)$. Second, we have $\mathbb{P}\{X_1 \geq X_2\} = \frac{e^{\mu_1}}{e^{\mu_1} + e^{\mu_2}}$. Third, letting $1(\cdot)$ be the indicator function, the random variables $\max\{X_1, X_2\}$ and $1(X_1 \geq X_2)$ are independent. The first and second properties are discussed in sections 7.2.2.2 and 7.2.2.3 and appendix B in Talluri and van Ryzin (2005). We show the third property in Online Appendix A. For a customer to purchase product $i \in S_k$, the customer's patience level must be at least k, the utility of the outside option must exceed the utilities of all products in stages $1, \ldots, k-1$, and the utility of product i must exceed both the utility of the outside option and the utilities of all other products in stage k. Letting $\hat{U}_{k-1} = \max_{j \in S_1 \cup \ldots \cup S_{k-1}} U_j$ and $\tilde{U}_k = \max_{j \in S_k \setminus \{i\}} U_j$, we have

$$\begin{aligned} \phi_i^k(S_1, \ldots, S_k) &= \mathbb{P}\{Y \geq k\} \cdot \mathbb{P}\Big\{U_0 \geq \max_{j \in S_1 \cup \ldots \cup S_{k-1}} U_j, \\ &\qquad \times U_i \geq \max\Big\{U_0, \max_{j \in S_k \setminus \{i\}} U_j\Big\}\Big\} \\ &= \lambda_k \cdot \mathbb{P}\{U_0 \geq \hat{U}_{k-1}\} \\ &\qquad \cdot \mathbb{P}\{U_i \geq \max\{U_0, \tilde{U}_k\} \mid U_0 \geq \hat{U}_{k-1}\}. \end{aligned} \quad (1)$$

By the first property, $\hat{U}_{k-1}$ and $\tilde{U}_k$ are Gumbel random variables with location-scale parameters $\left(\log\sum_{j\in S_1\cup\ldots\cup S_{k-1}} e^{\mu_j}, 1\right)$ and $\left(\log\sum_{j\in S_k\setminus\{i\}} e^{\mu_j}, 1\right)$. Moreover, the random variables U_i, U_0, $\hat{U}_{k-1}$, and $\tilde{U}_k$ are independent. Considering the second probability on the right side of (1), we have

$$\begin{aligned}
&\mathbb{P}\{U_i \ge \max\{U_0, \tilde{U}_k\} \mid U_0 \ge \hat{U}_{k-1}\}\\
&\quad = \mathbb{P}\{U_i \ge \max\{U_0, \tilde{U}_k, \hat{U}_{k-1}\} \mid U_0 \ge \hat{U}_{k-1}\}\\
&\quad = \mathbb{P}\{U_i \ge \max\{U_0, \hat{U}_{k-1}\},\ U_i \ge \tilde{U}_k \mid U_0 \ge \hat{U}_{k-1}\}\\
&\quad = \mathbb{P}\{U_i \ge \max\{U_0, \hat{U}_{k-1}\} \mid U_0 \ge \hat{U}_{k-1}\}\\
&\qquad \cdot\ \mathbb{P}\{U_i \ge \tilde{U}_k \mid U_i \ge \max\{U_0, \hat{U}_{k-1}\},\ U_0 \ge \hat{U}_{k-1}\}\\
&\quad \overset{(a)}{=} \mathbb{P}\{U_i \ge \max\{U_0, \hat{U}_{k-1}\}\}\\
&\qquad \cdot\ \mathbb{P}\{U_i \ge \tilde{U}_k \mid U_i \ge \max\{U_0, \hat{U}_{k-1}\},\ U_0 \ge \hat{U}_{k-1}\}\\
&\quad \overset{(b)}{=} \mathbb{P}\{U_i \ge \max\{U_0, \hat{U}_{k-1}\}\}\\
&\qquad \cdot\ \mathbb{P}\{U_i \ge \tilde{U}_k \mid U_i \ge \max\{U_0, \hat{U}_{k-1}\}\}\\
&\quad = \mathbb{P}\{U_i \ge \max\{U_0, \hat{U}_{k-1}\},\ U_i \ge \tilde{U}_k\}\\
&\quad = \mathbb{P}\{U_i \ge \max\{U_0, \tilde{U}_k, \hat{U}_{k-1}\}\}\\
&\quad \overset{(c)}{=} \frac{e^{\mu_i}}{1 + \sum_{j\in S_1\cup\ldots\cup S_k} e^{\mu_j}} = \frac{v_i}{1 + \sum_{\ell=1}^{k} V(S_\ell)}. \qquad (2)
\end{aligned}$$

Both (a) and (b) use the fact that $\max\{U_0, \hat{U}_{k-1}\}$ and $1(U_0 \ge \hat{U}_{k-1})$ are independent by the third property, so knowing that $U_0 \ge \hat{U}_{k-1}$ does not change the distribution of $\max\{U_0, \hat{U}_{k-1}\}$ along with U_i and $\tilde{U}_k$ independent of U_0 and $\hat{U}_{k-1}$. Finally, (c) uses the first and second properties.

Considering the first probability on the right side of (1), using the second property and the fact that $v_i = e^{\mu_i}$, we get $\mathbb{P}\{U_0 \ge \hat{U}_{k-1}\} = 1/\left(1 + \sum_{j\in S_1\cup\ldots\cup S_{k-1}} e^{\mu_j}\right) = 1/\left(1 + \sum_{\ell=1}^{k-1} V(S_\ell)\right)$. Plugging this equality and (2) into (1) gives the desired result. □

As an extension to our model, we can consider the case in which a customer, after not making a purchase in stage k, decides to continue to the next stage with probability β_k. If the decision to continue to the next stage is independent of $\{U_i : i \in \mathcal{N}\}$, U_0, and Y, then all we need to do is to multiply the choice probability in the theorem with $\beta_1\beta_2\ldots\beta_{k-1}$. Thus, this extension is equivalent to using a patience level distribution with $\mathbb{P}\{Y \ge k\} = \beta_1\beta_2\ldots\beta_{k-1}\lambda_k$.

3. Unconstrained Assortment Optimization

We focus on the assortment-optimization problem with no constraints on the offered assortment and give a polynomial-time algorithm. We use $r_i > 0$ to denote the revenue of product i. Throughout the paper, we let $W(S) = \sum_{i\in S} r_i v_i$. Noting the choice probability in Theorem 2.1, if we offer assortments $(S_1, \ldots, S_m)$ over m stages, then the expected revenue from a customer is

$$\begin{aligned}
&\Pi(S_1, \ldots, S_m)\\
&\quad = \sum_{k\in\mathcal{M}} \sum_{i\in S_k} r_i\, \phi_i^k(S_1, \ldots, S_m)\\
&\quad = \sum_{k\in\mathcal{M}} \sum_{i\in S_k} \frac{\lambda_k r_i v_i}{\left(1 + \sum_{\ell=1}^{k-1} V(S_\ell)\right)\left(1 + \sum_{\ell=1}^{k} V(S_\ell)\right)}\\
&\quad = \sum_{k\in\mathcal{M}} \frac{\lambda_k W(S_k)}{\left(1 + \sum_{\ell=1}^{k-1} V(S_\ell)\right)\left(1 + \sum_{\ell=1}^{k} V(S_\ell)\right)}. \qquad (3)
\end{aligned}$$

The assortments offered over m stages are disjoint, so the set of feasible solutions is $\mathcal{F} = \{(S_1, \ldots, S_m) : S_k \subseteq \mathcal{N}\ \ \forall k \in \mathcal{M},\ S_k \cap S_\ell = \varnothing\ \ \forall k \ne \ell\}$. We want to solve the problem

$$\max_{(S_1,\ldots,S_m)\in\mathcal{F}} \Pi(S_1, \ldots, S_m). \qquad (\text{Assortment})$$

We use two steps to give a polynomial-time algorithm for the (Assortment) problem. First, we show that there exists an optimal solution to the (Assortment) problem that is revenue-ordered. Specifically, we index the products in the order of decreasing revenues so that $r_1 \ge r_2 \ge \ldots \ge r_n$. Then, there exists an optimal solution $(S_1^*, \ldots, S_m^*)$ such that $S_k^* = \{j_k^* + 1, \ldots, j_{k+1}^*\}$ for $j_1^*, \ldots, j_{m+1}^*$ that satisfy $0 = j_1^* \le j_2^* \le \ldots \le j_{m+1}^*$. Thus, the assortment offered in each stage follows the order of the revenues of the products. Noting $j_1^* = 0$, the choice of the products $j_2^*, \ldots, j_{m+1}^*$ determines an optimal solution to the (Assortment) problem. Knowing that there exists an optimal solution that is revenue-ordered reduces the number of possible optimal solutions to $O(n^m)$, which is polynomial in n but still exponential in m. Second, exploiting the revenue-ordered property, we find an optimal sequence of revenue-ordered assortments by solving a dynamic program in $O(mn^2)$ operations.

3.1. Optimality of Revenue-Ordered Assortments

For two solutions $(S_1, \ldots, S_m)$ and $(T_1, \ldots, T_m)$, we say that the solution $(S_1, \ldots, S_m)$ dominates the solution $(T_1, \ldots, T_m)$ if $|S_1| = |T_1| \ldots |S_k| = |T_k|$ and $|S_{k+1}| > |T_{k+1}|$ for some $k \in \mathcal{M}$. Intuitively speaking, a dominating solution offers an assortment with a larger cardinality in an earlier stage. If there are multiple optimal solutions for the (Assortment) problem, then we choose an optimal solution that is nondominated by any other optimal solution. To establish that an optimal solution to the (Assortment) problem satisfies the revenue-ordered property, we construct revenue thresholds for each stage such that, if the revenue of a product falls within the thresholds for stage k, then it is optimal to offer the product in stage k. The next theorem is the main result of this section, establishing the existence of such revenue thresholds. The proof follows from an intermediate lemma, which we give after the theorem.

Theorem 3.1 (Optimal Revenue-Ordered Assortments). *There exists an optimal solution $(S_1^*, \ldots, S_m^*)$ to the (Assortment) problem such that $S_k^* = \{i \in \mathcal{N} : t_{k+1}^* \le r_i < t_k^*\}$ for some revenue thresholds $t_1^*, \ldots, t_{m+1}^*$ that satisfy $+\infty = t_1^* \ge t_2^* \ge \ldots \ge t_{m+1}^*$.*

To construct the revenue thresholds, let $R_k(S_1, \ldots, S_m) = \frac{\lambda_k W(S_k)}{\left(1+\sum_{\ell=1}^{k-1} V(S_\ell)\right)\left(1+\sum_{\ell=1}^{k} V(S_\ell)\right)}$ denote the expected revenue obtained in stage k, and define

$$t_k(S_1, \ldots, S_m) = \frac{R_{k-1}(S_1, \ldots, S_m) + R_k(S_1, \ldots, S_m)}{\frac{\lambda_{k-1}}{1+\sum_{\ell=1}^{k-2} V(S_\ell)} - \frac{\lambda_k}{1+\sum_{\ell=1}^{k} V(S_\ell)}} \quad \forall k \in \mathcal{M} \setminus \{1\},$$

$$t_{m+1}(S_1, \ldots, S_m) = \frac{R_m(S_1, \ldots, S_m)}{\frac{\lambda_m}{1+\sum_{\ell=1}^{m-1} V(S_\ell)}}.$$

We set $t_1(S_1, \ldots, S_m) = +\infty$. In the next lemma, we quantify the change in the expected revenue when we move a product from one stage to another. The proof is in Online Appendix B.

Lemma 3.1 (Product Exchanges). *For each sequence of assortments $(S_1, \ldots, S_m) \in \mathcal{F}$ offered over m stages, we have the following three identities:*

a. $$\Pi(S_1, \ldots, S_{k-1} \cup \{i\}, S_k \setminus \{i\}, \ldots, S_m) - \Pi(S_1, \ldots, S_m) = \frac{\frac{\lambda_{k-1}}{1+\sum_{\ell=1}^{k-2} V(S_\ell)} - \frac{\lambda_k}{1+\sum_{\ell=1}^{k} V(S_\ell)}}{1 + \sum_{\ell=1}^{k-1} V(S_\ell) + v_i} v_i (r_i - t_k(S_1, \ldots, S_m)) \quad \forall k = \mathcal{M} \setminus \{1\},\ i \in S_k,$$

b. $$\Pi(S_1, \ldots, S_k \setminus \{i\}, S_{k+1} \cup \{i\} \ldots, S_m) - \Pi(S_1, \ldots, S_m) = \frac{\frac{\lambda_k}{1+\sum_{\ell=1}^{k-1} V(S_\ell)} - \frac{\lambda_{k+1}}{1+\sum_{\ell=1}^{k+1} V(S_\ell)}}{1 + \sum_{\ell=1}^{k} V(S_\ell) - v_i} v_i (t_{k+1}(S_1, \ldots, S_m) - r_i) \quad \forall k \in \mathcal{M} \setminus \{m\},\ i \in S_k,$$

c. $$\Pi(S_1, \ldots, S_{m-1}, S_m \setminus \{i\}) - \Pi(S_1, \ldots, S_m) = \frac{\frac{\lambda_m}{1+\sum_{\ell=1}^{m-1} V(S_\ell)}}{1 + \sum_{\ell=1}^{m} V(S_\ell) - v_i} v_i (t_{m+1}(S_1, \ldots, S_m) - r_i) \quad \forall i \in S_m.$$

The proof of the lemma is based on directly evaluating the changes in the expected revenue using (3). Because $\lambda_k \ge \lambda_{k+1}$ and $\sum_{\ell=1}^{k} V(S_\ell) \le \sum_{\ell=1}^{k+1} V(S_\ell)$, by this lemma, we can compare r_i only with $t_k(S_1, \ldots, S_m)$ or $t_{k+1}(S_1, \ldots, S_m)$ to check whether moving product i from stage k to stage $k-1$ or to stage $k+1$ improves the expected revenue. The proof of Theorem 3.1 follows.

Proof of Theorem 3. 1. Let $(S_1^*, \ldots, S_m^*)$ be a nondominated optimal solution to the (Assortment) problem. Without loss of generality, $S_1^* \ne \varnothing, \ldots, S_\ell^* \ne \varnothing$, $S_{\ell+1}^* = \varnothing, \ldots, S_m^* = \varnothing$ for some $\ell \in \mathcal{M}$. In particular, if $S_\ell^* = \varnothing$ and $S_{\ell+1}^* \ne \varnothing$, then the solution $(S_1^*, \ldots, S_{\ell-1}^*, S_{\ell+1}^*, S_\ell^*, \ldots, S_m^*)$ dominates the solution $(S_1^*, \ldots, S_m^*)$, but because $\lambda_1 \ge \lambda_2 \ge \ldots \ge \lambda_m$, noting (3), we can check that the expected revenue of the solution $(S_1^*, \ldots, S_{\ell-1}^*, S_{\ell+1}^*, S_\ell^*, \ldots, S_m^*)$ is at least as large as that of the solution $(S_1^*, \ldots, S_m^*)$. Thus, because $S_{\ell+1}^* = \varnothing, \ldots, S_m^* = \varnothing$, a customer does not make a purchase after stage ℓ, so if we consider the (Assortment) problem with only ℓ stages, then $(S_1^*, \ldots, S_\ell^*)$ must be a nondominated optimal solution. For all $k = 1, \ldots, \ell$, we let $t_k^* = t_k(S_1^*, \ldots, S_\ell^*)$ and focus on the (Assortment) problem with ℓ stages, in which the set of stages is $\mathcal{L} = \{1, \ldots, \ell\}$.

First, we claim that $S_k^* \subseteq \{i \in \mathcal{N} : t_{k+1}^* \le r_i < t_k^*\}$ for all $k \in \mathcal{L}$. Specifically, for $k \in \mathcal{L} \setminus \{1\}$, if $i \in S_k^*$ and $r_i \ge t_k^*$, then Lemma 3.1(a) implies that moving product i from assortment S_k^* to S_{k-1}^* does not degrade the expected revenue from the solution $(S_1^*, \ldots, S_\ell^*)$, which contradicts the fact that $(S_1^*, \ldots, S_\ell^*)$ is a nondominated optimal solution. For $k = 1$, we cannot have $r_i \ge t_1^*$, because $t_1^* = +\infty$. For $k \in \mathcal{L} \setminus \{\ell\}$, if $i \in S_k^*$ and $r_i < t_{k+1}^*$, then Lemma 3.1(b) implies that moving product i from assortment S_k^* to S_{k+1}^* strictly increases the expected revenue from the solution $(S_1^*, \ldots, S_\ell^*)$, which contradicts the fact that $(S_1^*, \ldots, S_\ell^*)$ is an optimal solution. For $k = \ell$, if $i \in S_\ell^*$ and $r_i < t_{\ell+1}^*$, then Lemma 3.1(c) implies that removing product i from assortment S_ℓ^* strictly improves the expected revenue from the solution $(S_1^*, \ldots, S_\ell^*)$. So the claim holds. Also, for all $k \in \mathcal{L}$, $S_k^* \ne \varnothing$ includes some product i, so $t_{k+1}^* \le r_i < t_k^*$ for all $k \in \mathcal{L}$. Thus, $t_{k+1}^* \le t_k^*$ for all $k \in \mathcal{L}$.

Second, we claim that $S_k^* \supseteq \{i \in \mathcal{N} : t_{k+1}^* \le r_i < t_k^*\}$ for all $k \in \mathcal{L}$. Specifically, if $t_{k+1}^* \le r_i < t_k^*$ and $i \notin S_k^*$ for some $k \in \mathcal{L}$, then it must be the case that $i \notin S_q^*$ for all $q \in \mathcal{L}$ because, by the first claim, we have $S_q^* \subseteq \{i \in \mathcal{N} : t_{q+1}^* \le r_i < t_q^*\}$ for all $q \in \mathcal{L}$ and $+\infty = t_1^* \ge t_2^* \ge \cdots \ge t_{\ell+1}^*$. In this case, using the fact that $r_i \ge t_{k+1}^* \ge t_{\ell+1}^*$, replacing the preference weight of product i in Lemma 3.1(c) with $-v_i$, note that adding product i to assortment S_ℓ^* does not degrade the expected revenue from the solution $(S_1^*, \ldots, S_\ell^*)$, which contradicts the fact that $(S_1^*, \ldots, S_\ell^*)$ is a nondominated optimal solution to the (Assortment) problem with ℓ stages. So the claim holds. By the two claims, we get $S_k^* = \{i \in \mathcal{N} : t_{k+1}^* \le r_i < t_k^*\}$ for all $k \in \mathcal{L}$ and $+\infty = t_1^* \ge t_2^* \ge \ldots \ge t_{\ell+1}^*$. For the problem with m stages, noting that $S_k^* = \varnothing$ for all $k = \ell+1, \ldots, m$, we set $t_k^* = t_{\ell+1}^*$ for all $k = \ell+2, \ldots,\ m+1$, so we have

$S_k^* = \{i \in \mathcal{N} : t_{k+1}^* \le r_i < t_k^*\}$ for all $k \in \mathcal{M}$ and $+\infty = t_1^* \ge t_2^* \ge \ldots \ge t_{m+1}^*$. □

3.2. Finding an Optimal Sequence of Revenue-Ordered Assortments

We show how to find an optimal sequence of revenue-ordered assortments. By Theorem 3.1, we can consider solutions $(S_1, \ldots, S_m)$ of the form $S_k = \{j_k + 1, \ldots, j_{k+1}\}$ for $j_1, \ldots, j_{m+1}$ that satisfy $0 = j_1 \le j_2 \le \ldots \le j_{m+1}$. If we offer assortments of this form, then $S_1 \cup \ldots \cup S_{k-1} = \{1, \ldots, j_k\}$ and $\sum_{\ell=1}^{k-1} V(S_\ell) = V(\{1, \ldots, j_k\})$. Thus, we can solve a dynamic program to pick assortments of this form to offer in each stage so that we maximize the expected revenue. The decision epochs are the stages. The state variable at decision epoch k is the value of j such that the assortments $S_1, \ldots, S_{k-1}$ offered in the previous stages satisfy $S_1 \cup \ldots \cup S_{k-1} = \{1, \ldots, j\}$. The action at decision epoch k is the value of p such that the assortment offered in stage k is $\{j+1, \ldots, p\}$. Let $J_k(j)$ denote the maximum expected revenue obtained from stages $k, k+1, \ldots, m$ given that $S_1 \cup \ldots \cup S_{k-1} = \{1, \ldots, j\}$. The next theorem gives a dynamic programming formulation to compute $\{J_k(j) : j \in \mathcal{N},\ k \in \mathcal{M}\}$.

Theorem 3.2 (Dynamic Program for an Optimal Sequence of Assortments). *Letting $J_{m+1}(\cdot) = 0$, for all $k \in \mathcal{M}$ and $j \in \mathcal{N}$, we have*

$$J_k(j) = \max_{p \in \{j, \ldots, n\}} \left\{ \frac{\lambda_k W\Big(\{j+1, \ldots, p\}\Big)}{\Big(1 + V\Big(\{1, \ldots, j\}\Big)\Big)\Big(1 + V\Big(\{1, \ldots, p\}\Big)\Big)} + J_{k+1}(p) \right\},$$

and we can solve the dynamic program in $O(mn^2)$ operations.

Proof. The dynamic program follows from the preceding discussion. Precomputing $W(\{j+1, \ldots, p\})$ and $V(\{1, \ldots, j\})$ for all $j, p \in \mathcal{N}$ with $p > j$ in $O(n^2)$ operations because there are m decision epochs, n states, and n actions, we can solve the dynamic program in $O(mn^2)$ operations. □

4. Joint Pricing and Assortment Optimization

We consider the joint pricing and assortment-optimization problem, in which we choose the assortment of products to offer in each stage as well as the prices of the products.

4.1. Optimal Prices Under Fixed Assortments

In this section, we assume that the assortments $(S_1, \ldots, S_m)$ offered over m stages are fixed. We give a convex program to choose the prices to maximize the expected revenue. In Section 4.2, we build on this result to give an approximation algorithm for joint pricing and assortment optimization. We use p_i to denote the price for product i. For fixed parameters α_i and β, if we charge the price p_i for product i, then the utility of product i has the Gumbel distribution with location-scale parameters $(\alpha_i - \beta p_i, 1)$ with the corresponding mean $\alpha_i - \beta p_i + \gamma$, where γ is the Euler–Mascheroni constant. Thus, the mean utility of a product depends linearly on its price. Such linear dependence of the mean utility on the price is often used in the literature; see Song and Xue (2007), Gallego and Wang (2014), and Li and Webster (2017). The parameter α_i captures the intrinsic mean utility of product i, whereas β captures the sensitivity of the mean utility to price. If we charge the price p_i for product i, then the preference weight of the product is $e^{\alpha_i - \beta p_i}$. As a function of the prices $\boldsymbol{p} = (p_1, \ldots, p_n)$, let $V_k(\boldsymbol{p}) = \sum_{i \in S_k} e^{\alpha_i - \beta p_i}$ capture the total preference weight of the products in stage k. Because the assortment of products offered in each stage is fixed, we do not make the dependence of $V_k(\boldsymbol{p})$ on the assortment S_k explicit. By Theorem 2.1, if the prices of the products are given by $\boldsymbol{p}$, then a customer chooses product $i \in S_k$ with probability $\phi_i^k(\boldsymbol{p}) = \lambda_k e^{\alpha_i - \beta p_i} / \Big(\Big(1 + \sum_{\ell=1}^{k-1} V_\ell(\boldsymbol{p})\Big)\Big(1 + \sum_{\ell=1}^{k} V_\ell(\boldsymbol{p})\Big)\Big)$. As a function of the prices $\boldsymbol{p}$, the expected revenue obtained from a customer is

$$\Pi(\boldsymbol{p}) = \sum_{k \in \mathcal{M}} \sum_{i \in S_k} p_i \phi_i^k(\boldsymbol{p}) = \sum_{k \in \mathcal{M}} \frac{\lambda_k \sum_{i \in S_k} p_i e^{\alpha_i - \beta p_i}}{\Big(1 + \sum_{\ell=1}^{k-1} V_\ell(\boldsymbol{p})\Big)\Big(1 + \sum_{\ell=1}^{k} V_\ell(\boldsymbol{p})\Big)}.$$

In the pricing literature, it is customary to include a marginal cost c_i for product i so that the objective function presented reads $\sum_{k \in \mathcal{M}} \sum_{i \in S_k} (p_i - c_i) \phi_i^k(\boldsymbol{p})$. Including a marginal cost for product i is equivalent to simply shifting the price of product i by c_i and the constant α_i by βc_i. We want to find the product prices to maximize the expected revenue, yielding the problem

$$\max_{\boldsymbol{p} \in \mathbb{R}^n} \ \Pi(\boldsymbol{p}). \qquad \text{(PRICING)}$$

The prices are not constrained to be nonnegative, allowing us to use first-order conditions to characterize an optimal solution. In Online Appendix C, we show that the optimal prices are nonnegative.

4.1.1. Stage-Specific Optimal Prices.

The following theorem shows that the prices in a particular stage are the same in an optimal solution. We use this result to give a convex program to solve the (PRICING) problem.

Theorem 4.1 (Stage-Specific Optimal Prices). *There exists an optimal solution $\boldsymbol{p}^*$ to the* (Pricing) *problem such that, if $i,j \in S_\ell$ for some $\ell \in \mathcal{M}$, then $p_i^* = p_j^*$.*

Proof. Letting $\boldsymbol{p}^*$ be an optimal solution to the (Pricing) problem, assume that $p_i^* \neq p_j^*$ for some $\ell \in \mathcal{M}$ and $i,j \in S_\ell$. We construct another solution $\hat{\boldsymbol{p}}$ with $\hat{p}_i = \hat{p}_j$ and $\hat{p}_t = p_t^*$ for all $t \in \mathcal{N} \setminus \{i,j\}$ such that the solution $\hat{\boldsymbol{p}}$ provides an expected revenue that is at least as large as the one provided by the solution $\boldsymbol{p}^*$. Specifically, set $\hat{p}_t = p_t^*$ for all $t \in \mathcal{N} \setminus \{i,j\}$. Thus, letting $W_k(\boldsymbol{p}) = \sum_{t \in S_k} p_t \, e^{\alpha_t - \beta p_t}$, we have $V_k(\hat{\boldsymbol{p}}) = V_k(\boldsymbol{p}^*)$ and $W_k(\hat{\boldsymbol{p}}) = W_k(\boldsymbol{p}^*)$ for all $k \in \mathcal{M} \setminus \{\ell\}$. Letting $K^* = e^{\alpha_i - \beta p_i^*} + e^{\alpha_j - \beta p_j^*}$, set $(\hat{p}_i, \hat{p}_j)$ as an optimal solution to the problem

$$\max_{(p_i,p_j)\in\mathbb{R}^2} \left\{ p_i \, e^{\alpha_i - \beta p_i} + p_j \, e^{\alpha_j - \beta p_j} \; : \; e^{\alpha_i - \beta p_i} + \; e^{\alpha_j - \beta p_j} = K^* \right\}. \tag{4}$$

Because (p_i^*, p_j^*) is a feasible solution to Problem (4) and $\hat{p}_t = p_t^*$ for all $t \in S_\ell \setminus \{i,j\}$, we have $W_\ell(\hat{\boldsymbol{p}}) \geq W_\ell(\boldsymbol{p}^*)$. Also, noting the constraint, we have $e^{\alpha_i - \beta \hat{p}_i} + e^{\alpha_j - \beta \hat{p}_j} = K^* = e^{\alpha_i - \beta p_i^*} + e^{\alpha_j - \beta p_j^*}$, so $V_\ell(\boldsymbol{p}^*) = V_\ell(\hat{\boldsymbol{p}})$. In this case, because $V_k(\hat{\boldsymbol{p}}) = V_k(\boldsymbol{p}^*)$ and $W_k(\hat{\boldsymbol{p}}) = W_k(\boldsymbol{p}^*)$ for all $k \in \mathcal{M} \setminus \{\ell\}$, noting the definition of $\Pi(\boldsymbol{p})$, we get $\Pi(\hat{\boldsymbol{p}}) \geq \Pi(\boldsymbol{p}^*)$ as desired. It remains to show that the optimal solution $(\hat{p}_i, \hat{p}_j)$ to Problem (4) satisfies $\hat{p}_i = \hat{p}_j$. Using the change of variables $d_i = e^{\alpha_i - \beta p_i}$ and solving for p_i, we have $p_i = \frac{1}{\beta}(\alpha_i - \log d_i)$. Thus, Problem (4) is equivalent to the problem $\frac{1}{\beta} \max_{(d_i,d_j)\in\mathbb{R}_+^2} \{(\alpha_i - \log d_i)\, d_i + (\alpha_j - \log d_j)\, d_j \; : \; d_i + d_j = K^*\}$. Because $-x \log x$ is concave in x, we can solve the last problem using Lagrangian relaxation. Associating the Lagrange multiplier θ with the constraint, the Lagrangian is $(\alpha_i - \log d_i)\, d_i + (\alpha_j - \log d_j)\, d_j + \theta(K^* - d_i - d_j)$. Differentiating the Lagrangian, the optimal solution $(\hat{d}_i, \hat{d}_j)$ to the last problem satisfies $\alpha_i - \log \hat{d}_i - 1 - \theta = 0$ and $\alpha_j - \log \hat{d}_j - 1 - \theta = 0$, so $\alpha_i - \log \hat{d}_i = 1 + \theta = \alpha_j - \log \hat{d}_j$. Recalling that $p_i = \frac{1}{\beta}(\alpha_i - \log d_i)$, the optimal solution $(\hat{p}_i, \hat{p}_j)$ to (4) satisfies $\hat{p}_i = \frac{1}{\beta}(\alpha_i - \log \hat{d}_i) = \frac{1}{\beta}(1 + \theta) = \frac{1}{\beta}(\alpha_j - \log \hat{d}_j) = \hat{p}_j$. □

By the preceding theorem, we can focus on the solutions in which all products in each stage have the same price. Next, we give a convex reformulation of the (Pricing) problem.

4.1.2. Convex Reformulation of the Pricing Problem.

By Theorem 4.1, letting ρ_k be the price that we charge for all products in stage k, we can use stage-specific prices $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_m)$ instead of product-specific prices $\boldsymbol{p} = (p_1, \ldots, p_n)$ as the decision variables. It is simple to give examples to demonstrate that the expected revenue is not a concave function of either $\boldsymbol{p}$ or $\boldsymbol{\rho}$. We give an equivalent formulation for the (Pricing) problem, which has a concave objective function and linear constraints. Using stage-specific prices $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_m)$, the total preference weight of the products in stage k is $\hat{V}_k(\boldsymbol{\rho}) = \sum_{i \in S_k} e^{\alpha_i - \beta \rho_k} = e^{-\beta \rho_k} \sum_{i \in S_k} e^{\alpha_i}$. Noting the definition of the purchase probability for product i in Theorem 2.1, if we charge the stage-specific prices $\boldsymbol{\rho}$, then the probability that a customer purchases some product in stage k is $\frac{\lambda_k \sum_{i \in S_k} e^{\alpha_i - \beta \rho_k}}{\left(1 + \sum_{\ell=1}^{k-1} \hat{V}_\ell(\boldsymbol{p})\right)\left(1 + \sum_{\ell=1}^{k} \hat{V}_\ell(\boldsymbol{p})\right)} = \frac{\lambda_k \hat{V}_k(\boldsymbol{\rho})}{\left(1 + \sum_{\ell=1}^{k-1} \hat{V}_\ell(\boldsymbol{p})\right)\left(1 + \sum_{\ell=1}^{k} \hat{V}_\ell(\boldsymbol{p})\right)}$, in which case, the price of the purchased product is ρ_k. Throughout the rest of this section, we let $q_k(\boldsymbol{\rho}) = 1 / \left(1 + \sum_{\ell=1}^{k} \hat{V}_\ell(\boldsymbol{\rho})\right)$ with the convention that $q_0(\boldsymbol{\rho}) = 1$. For each $k \in \mathcal{M}$, $q_k(\boldsymbol{\rho})$ is the probability that the utility of the outside option exceeds the utility of all products offered in the first k stages.

We refer to $q_k(\boldsymbol{\rho})$ as the no-purchase probability over the first k stages, but we understand that this probability is actually the no-purchase probability for a customer with patience level exceeding k. The idea behind our convex reformulation is to express the probability that a customer makes a purchase in each stage and the stage-specific price for each stage as functions of the no-purchase probabilities over different numbers of stages. By doing so, we express the expected revenue as a function of the no-purchase probabilities over different numbers of stages as well. Specifically, noting that $q_k(\boldsymbol{\rho}) = \frac{1}{1 + \sum_{\ell=1}^{k} \hat{V}_\ell(\boldsymbol{\rho})}$, we have $q_{k-1}(\boldsymbol{\rho}) - q_k(\boldsymbol{\rho}) = \frac{\hat{V}_k(\boldsymbol{\rho})}{\left(1 + \sum_{\ell=1}^{k-1} \hat{V}_\ell(\boldsymbol{\rho})\right)\left(1 + \sum_{\ell=1}^{k} \hat{V}_\ell(\boldsymbol{\rho})\right)}$. Therefore, given the no-purchase probabilities $\boldsymbol{q} = (q_1, \ldots, q_m)$ over different numbers of stages, the probability that a customer purchases a product in stage k is $\lambda_k (q_{k-1} - q_k)$. Moreover, using the fact that $q_k(\boldsymbol{\rho}) = \frac{1}{1 + \sum_{\ell=1}^{k} \hat{V}_\ell(\boldsymbol{\rho})}$, we get $\frac{1}{q_k(\boldsymbol{\rho})} - \frac{1}{q_{k-1}(\boldsymbol{\rho})} = \hat{V}_k(\boldsymbol{\rho}) = e^{-\beta \rho_k} \sum_{i \in S_k} e^{\alpha_i}$. In this case, solving for ρ_k in the equality $\frac{1}{q_k(\boldsymbol{\rho})} - \frac{1}{q_{k-1}(\boldsymbol{\rho})} = e^{-\beta \rho_k} \sum_{i \in S_k} e^{\alpha_i}$ given the no-purchase probabilities $\boldsymbol{q} = (q_1, \ldots, q_m)$ over different numbers of stages, the stage-specific price for stage k is

$$\rho_k(\boldsymbol{q}) = \frac{1}{\beta} \left\{ \log\left(\sum_{i \in S_k} e^{\alpha_i} \right) - \log\left(\frac{1}{q_k} - \frac{1}{q_{k-1}} \right) \right\}.$$

Thus, for given no-purchase probabilities $\boldsymbol{q} = (q_1, \ldots, q_m)$, the customer makes a purchase in stage k with probability $\lambda_k (q_{k-1} - q_k)$. If the customer does so, then the price of the purchased product is $\rho_k(\boldsymbol{q})$.

By the preceding discussion, we can express the expected revenue as a function of the no-purchase probabilities. That is, we have

$$\hat{\Pi}(\boldsymbol{q}) = \sum_{k\in\mathcal{M}} \lambda_k\,(q_{k-1}-q_k)\;\rho_k(\boldsymbol{q}) = \sum_{k\in\mathcal{M}} \frac{\lambda_k}{\beta}\,(q_{k-1}-q_k) \times \left\{\log\left(\sum_{i\in S_k} e^{\alpha_i}\right) - \log\left(\frac{1}{q_k}-\frac{1}{q_{k-1}}\right)\right\}. \tag{5}$$

In the next theorem, we show that this expected revenue function is concave in the no-purchase probabilities and we can recover the optimal prices using its maximizer.

Theorem 4.2 (Convex Reformulation for Pricing). *The expected revenue $\hat{\Pi}(\boldsymbol{q})$ in (5) is a concave function of $\boldsymbol{q}$. Furthermore, letting $\boldsymbol{q}^*$ be an optimal solution to the problem*

$$\max_{\boldsymbol{q}\in\mathbb{R}^m}\left\{\hat{\Pi}(\boldsymbol{q})\;:\; q_{k-1}\ge q_k \quad \forall\, k\in\mathcal{M}\right\} \tag{6}$$

with the convention that $q_0 = 1$, if we set $\rho_k^* = \rho_k(\boldsymbol{q}^*)$ for all $k\in\mathcal{M}$, then $\boldsymbol{\rho}^*$ are optimal stage-specific prices to charge in the (Pricing) *problem.*

Proof. To show that $\hat{\Pi}(\boldsymbol{q})$ is a concave function of $\boldsymbol{q}$, noting that $\sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log\sum_{i\in S_k}e^{\alpha_i}$ is linear in $\boldsymbol{q}$, it suffices to prove that $\sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log\left(\frac{1}{q_k}-\frac{1}{q_{k-1}}\right)$ is convex in $\boldsymbol{q}$. We have $\sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log\left(\frac{1}{q_k}-\frac{1}{q_{k-1}}\right) = \sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\log\frac{q_{k-1}-q_k}{q_{k-1}} - \sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log q_k$. First, we show that $\sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log\frac{q_{k-1}-q_k}{q_{k-1}}$ is convex in $\boldsymbol{q}$. The relative entropy function $x\log(x/y)$ is convex in $(x,y)\in\mathbb{R}_+^2$; see example 3.19 in Boyd and Vandenberghe (2004). Moreover, composing a convex function with an affine function preserves its convexity; see section 3.2.2 in Boyd and Vandenberghe (2004). Thus, $(q_{k-1}-q_k)\,\log\frac{q_{k-1}-q_k}{q_{k-1}}$ is convex in $\boldsymbol{q}$, in which case, $\sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log\frac{q_{k-1}-q_k}{q_{k-1}}$ is convex in $\boldsymbol{q}$. Second, we show that $-\sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log q_k$ is convex in $\boldsymbol{q}$. Noting that $q_0 = 1$ and rearranging the terms in the sum, we have

$$\begin{aligned}
-\sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log q_k &= -\lambda_1\,\log q_1 \\
&\quad + \sum_{k=1}^{m-1} q_k\,(\lambda_k\log q_k - \lambda_{k+1}\,\log q_{k+1}) + \lambda_m\,q_m\,\log q_m \\
&= -\lambda_1\,\log q_1 + \sum_{k=1}^{m-1}\lambda_{k+1}\,q_k(\log q_k - \log q_{k+1}) \\
&\quad + \sum_{k=1}^{m-1}(\lambda_k-\lambda_{k+1})\,q_k\,\log q_k + \lambda_m\,q_m\,\log q_m.
\end{aligned}$$

Because $x\log(x/y)$ and $x\log x$ are convex in $(x,y)\in\mathbb{R}_+^2$ and $\lambda_k\ge\lambda_{k+1}$, $-\sum_{k\in\mathcal{M}}\lambda_k\,(q_{k-1}-q_k)\,\log q_k$ is convex in $\boldsymbol{q}$. The second part of the theorem holds by the discussion before the theorem. □

Problem (6) has a concave objective function and linear constraints, so we can solve it efficiently using convex optimization tools. Also, by (5), once we fix the values of $(q_1,\ldots,q_{k-1})$, the optimal values of $(q_k,\ldots,q_m)$ only depend on q_{k-1}. In Online Appendix D, using this observation, we give a dynamic program to find a solution with an additive performance guarantee of θ with a running time polynomial in $1/\theta$. Numerically, solving Problem (6) through convex optimization tools turns out to be faster, but the dynamic program does not require convex optimization software.

4.1.3. Monotonicity of Optimal Prices.

In the next theorem, we compare the optimal prices in different stages. If $\lambda_k = 1$ for all $k\in\mathcal{M}$, then the patience level of the customers is m with probability one, which is to say that the customers leave the system only when they have found a product with utility exceeding the utility of the outside option or they have exhausted all stages and still have not found a product with utility exceeding the utility of the outside option. By the next theorem, if $\lambda_k = 1$ for all $k\in\mathcal{M}$, then the optimal prices in stage k are at least as large as those in stage $k+1$ for each $k\in\mathcal{M}\setminus\{m\}$.

Theorem 4.3 (Monotonicity of Scaled Prices). *There exist optimal stage-specific prices $\boldsymbol{\rho}^* = (\rho_1^*,\ldots,\rho_m^*)$ in the* (Pricing) *problem such that $\lambda_k\,\rho_k^*\ge\lambda_{k+1}\,\rho_{k+1}^*$ for all $k = 1,\ldots,m-1$.*

Proof. Letting $q_k^* = q_k(\boldsymbol{\rho}^*)$ for all $k\in\mathcal{M}$ with $q_0^* = 1$, in Online Appendix E, we use the first-order condition for the (Pricing) problem to show that the optimal stage-specific prices satisfy

$$\frac{1}{\beta} - \frac{q_\ell^*}{q_{\ell-1}^*}\,\rho_\ell^* + \frac{1}{\lambda_\ell\,q_\ell^*\,q_{\ell-1}^*}\sum_{k=\ell+1}^{m}\rho_k^*\,\lambda_k\left\{(q_{k-1}^*)^2 - (q_k^*)^2\right\} = 0. \tag{7}$$

Letting $Q_{\ell+1}^* = \sum_{k=\ell+1}^{m}\rho_k^*\,\lambda_k\,\left((q_{k-1}^*)^2 - (q_k^*)^2\right)$, (7) reads as $\frac{q_\ell^*}{q_{\ell-1}^*}\,\rho_\ell^* = \frac{1}{\beta} + \frac{1}{\lambda_\ell\,q_\ell^*\,q_{\ell-1}^*}\,Q_{\ell+1}^*$. Similarly, using (7) for stage $\ell+1$, we have $\frac{q_{\ell+1}^*}{q_\ell^*}\,\rho_{\ell+1}^* = \frac{1}{\beta} + \frac{1}{\lambda_{\ell+1}\,q_{\ell+1}^*\,q_\ell^*}\,Q_{\ell+2}^*$, which is equivalent to

$$\frac{1}{\lambda_\ell\,q_\ell^*\,q_{\ell-1}^*}\,Q_{\ell+2}^* = \frac{\lambda_{\ell+1}\,(q_{\ell+1}^*)^2}{\lambda_\ell\,q_\ell^*\,q_{\ell-1}^*}\,\rho_{\ell+1}^* - \frac{\lambda_{\ell+1}\,q_{\ell+1}^*}{\beta\,\lambda_\ell\,q_{\ell-1}^*}. \tag{8}$$

Noting the definition of $Q_{\ell+1}^*$, we have $Q_{\ell+1}^* = \rho_{\ell+1}^*\,\lambda_{\ell+1}\,((q_\ell^*)^2 - (q_{\ell+1}^*)^2) + Q_{\ell+2}^*$. Therefore, using the fact that $\frac{q^*_\ell}{q^*_{\ell-1}}\,\rho^*_\ell = \frac{1}{\beta} + \frac{1}{\lambda_\ell\, q^*_\ell\, q^*_{\ell-1}}\,Q^*_{\ell+1}$, we obtain the chain of equalities

$$\begin{aligned}\frac{q^*_\ell}{q^*_{\ell-1}}\,\rho^*_\ell &= \frac{1}{\beta} + \frac{1}{\lambda_\ell\, q^*_\ell\, q^*_{\ell-1}} \\ &\qquad \times \left\{\rho^*_{\ell+1}\,\lambda_{\ell+1}\left((q^*_\ell)^2 - (q^*_{\ell+1})^2\right) + Q^*_{\ell+2}\right\} \\ &\overset{(a)}{=} \frac{1}{\beta} + \frac{1}{\lambda_\ell\, q^*_\ell\, q^*_{\ell-1}}\,\rho^*_{\ell+1}\,\lambda_{\ell+1}\left((q^*_\ell)^2 - (q^*_{\ell+1})^2\right) \\ &\qquad + \frac{\lambda_{\ell+1}\,(q^*_{\ell+1})^2}{\lambda_\ell\, q^*_\ell\, q^*_{\ell-1}}\,\rho^*_{\ell+1} - \frac{\lambda_{\ell+1}\, q^*_{\ell+1}}{\beta\,\lambda_\ell\, q^*_{\ell-1}} \\ &= \left(1 - \frac{\lambda_{\ell+1}\, q^*_{\ell+1}}{\lambda_\ell\, q^*_{\ell-1}}\right)\frac{1}{\beta} + \frac{\lambda_{\ell+1}\, q^*_\ell}{\lambda_\ell\, q^*_{\ell-1}}\,\rho^*_{\ell+1},\end{aligned}$$

where (a) uses (8). By this chain of equalities, $\frac{q^*_\ell}{\lambda_\ell\, q^*_{\ell-1}}(\lambda_\ell\,\rho^*_\ell - \lambda_{\ell+1}\,\rho^*_{\ell+1}) = (1 - \frac{\lambda_{\ell+1}\, q^*_{\ell+1}}{\lambda_\ell\, q^*_{\ell-1}})\frac{1}{\beta}$, so because $q^*_{\ell+1} \le q^*_{\ell-1}$ and $\lambda_{\ell+1} \le \lambda_\ell$, we get $\lambda_\ell\,\rho^*_\ell \ge \lambda_{\ell+1}\,\rho^*_{\ell+1}$. □

Our results in this section use the assumption that the products have the same price sensitivity, which is reasonable for products in the same product category. When studying pricing problems even under the standard multinomial logit model with a single stage, it is common to use the assumption that the products have the same price sensitivity. For example, Hopp and Xu (2005), Song and Xue (2007), Li and Huh (2011), and Zhang and Lu (2013) use this assumption when working with the standard multinomial logit model. These papers use market shares of the products as the decision variables to give convex reformulations, whereas we use the no-purchase probabilities. When different products have different price sensitivities, we are not aware of convex reformulations of the pricing problem even under the standard multinomial logit model, but it is not known whether such convex reformulations provably do not exist.

4.2. Optimal Assortments and Prices

In this section, we consider the case in which both the assortments $(S_1, \ldots, S_m)$ offered over m stages and the prices charged for the products are decision variables. By the discussion in the previous section, for any fixed sequence of assortments, it is optimal to charge stage-specific prices. Therefore, it suffices to focus on stage-specific prices when both the sequence of assortments to offer and the prices to charge are decision variables, but to simplify the proofs of our results, we revert to using product-specific prices. As a function of the product-specific prices $\boldsymbol{p} = (p_1, \ldots, p_n)$, we let $V(\boldsymbol{p}, S) = \sum_{i\in S} e^{\alpha_i - \beta p_i}$ to capture the total preference weight of the products in S. Noting Theorem 2.1, if we charge the prices $\boldsymbol{p}$ and offer the assortments $(S_1, \ldots, S_m)$, then a customer purchases product $i \in S_k$ with probability $\lambda_k\, e^{\alpha_i - \beta p_i} / \left(\left(1 + \sum_{\ell=1}^{k-1} V(\boldsymbol{p}, S_\ell)\right)\left(1 + \sum_{\ell=1}^{k} V(\boldsymbol{p}, S_\ell)\right)\right)$. Thus, as a function of the product-specific prices $\boldsymbol{p}$ and the assortments $(S_1, \ldots, S_m)$ over m stages, the expected revenue is

$$\Pi(\boldsymbol{p}, S_1, \ldots, S_m) = \sum_{k\in\mathcal{M}} \frac{\lambda_k \sum_{i\in S_k} p_i\, e^{\alpha_i - \beta p_i}}{\left(1 + \sum_{\ell=1}^{k-1} V(\boldsymbol{p}, S_\ell)\right) \times \left(1 + \sum_{\ell=1}^{k} V(\boldsymbol{p}, S_\ell)\right)}.$$

We continue to use $\mathcal{F}$ to denote the set of feasible assortments that we can offer over m stages, ensuring that the assortments offered over different stages are disjoint.

Our goal is to find the assortment to offer in each stage and the prices to charge for the products to maximize the expected revenue, yielding the problem

$$\max_{(\boldsymbol{p}, S_1, \ldots, S_m)\in\mathbb{R}^n\times\mathcal{F}} \Pi(\boldsymbol{p}, S_1, \ldots, S_m). \tag{Pricing-Assortment}$$

This problem involves both continuous and discrete decision variables. In the rest of this section, we focus on obtaining solutions with performance guarantees for this problem.

4.2.1. Approximation Algorithm for Joint Pricing and Assortment Optimization. In the next theorem, we show that, if we offer all products simply in the first stage and compute the corresponding optimal prices, then we obtain a 0.878-approximate solution. By Theorem 4.2, we can solve a convex program to compute the optimal prices for a fixed sequence of assortments.

Theorem 4.4 (87.8% Approximation for Joint Pricing and Assortment Optimization). *Letting π^* be the optimal objective value of the* (Pricing-Assortment) *problem, we have* $\max_{\boldsymbol{p}\in\mathbb{R}^n} \Pi(\boldsymbol{p}, \mathcal{N}, \emptyset, \ldots, \emptyset) \ge 0.878\,\pi^*$.

In Online Appendix F, we give a proof for Theorem 4.4 and show that the performance guarantee of 87.8% is tight. The proof is based on a sequence of upper bounds. First, we consider a variant of the (Pricing-Assortment) problem in which the patience levels of the customers are, intuitively speaking, infinite. We argue that the optimal expected revenue of the variant with infinite patience levels provides an upper bound on that of the (Pricing-Assortment) problem. Second, by treating $\left(\sum_{i\in S_1} e^{\alpha_i}, \ldots, \sum_{i\in S_m} e^{\alpha_i}\right)$ in (5) as continuous quantities, we formulate a smooth variant of the (Pricing-Assortment) problem. We argue that the optimal expected revenue of the smooth variant is an upper bound on that of the variant with infinite patience levels. Third, we give a closed-form upper bound on the optimal expected revenue of the smooth variant. Chaining all upper bounds, the closed-form upper bound is also an upper bound on the optimal expected revenue of the (Pricing-Assortment) problem. Finally, the closed-form upper bound is simple enough to allow us to show that the expected revenue obtained by offering all products in the first stage and computing the corresponding optimal prices is at least 87.8% of the closed-form upper bound.

Theorem 4.4 also allows us to make an interesting contrast between the (Assortment) and (Pricing-Assortment) problems. For the (Assortment) problem, we can show that, if we offer the empty assortment in all stages except for the first stage and find the revenue-maximizing assortment to offer in the first stage, then we obtain a solution that provides at least 50% of the optimal expected revenue in the (Assortment) problem. In Online Appendix G, we give a proof for the performance guarantee of 50% and show that it is tight. Thus, for the (Pricing-Assortment) problem, finding the revenue-maximizing prices to charge in the first stage while offering the empty assortment in all other stages yields a tight performance guarantee of 87.8%, whereas for the (Assortment) problem, finding the revenue-maximizing assortment to offer in the first stage while offering the empty assortment in all other stages yields a tight performance guarantee of 50%.

In our computational experiments, we use a neighborhood search algorithm to further improve the performance of the solution obtained by offering all products in the first stage and computing the corresponding optimal prices. In particular, we start with a sequence of assortments that offers all products in the first stage. Given the current sequence of assortments, we check all neighbors of the current sequence of assortments for an appropriately defined neighborhood. For each sequence of assortments in the neighborhood, we compute the corresponding optimal prices to charge. Among all sequences of assortments in the neighborhood and their corresponding optimal prices, we pick the best one. We repeat the process starting from the best sequence of assortments in the neighborhood until we cannot improve the expected revenue. This algorithm is guaranteed to provide a solution that is at least as good as the solution that we start with.

In this section, we gave an approximation algorithm for the (Pricing-Assortment) problem. In this problem, the prices take values over a continuum and the preference weight of product i is given by $e^{\alpha_i-\beta p_i}$ as a function of its price p_i. A natural question is the computational complexity of this problem. In Online Appendix H, we show that, if the prices take values over a discrete set, then the (Pricing-Assortment) problem is NP-hard, but the computational complexity of the problem with the prices taking values over a continuum remains an open question.

5. Assortment Optimization Under a Space Constraint

We consider the assortment problem when each product occupies a certain amount of space and there is a limit on the total space consumption of the products offered in all stages. As in Section 3, the revenue of product i is r_i, and we index the products such that $r_1 \geq r_2 \geq \ldots \geq r_n$. The space consumption of product i is c_i. We let $C(S) = \sum_{i\in S} c_i$. The total amount of space available is b. Noting the expected revenue function in the (Assortment) problem, we want to solve

$$\max_{(S_1,\ldots,S_m)\in\mathcal{F}} \left\{ \sum_{k\in\mathcal{M}} \frac{\lambda_k W(S_k)}{\left(1+\sum_{q=1}^{k-1} V(S_q)\right)\times\left(1+\sum_{q=1}^{k} V(S_q)\right)} : \sum_{k\in\mathcal{M}} C(S_k) \leq b \right\}. \tag{Capacitated}$$

5.1. Overview of Our Approach

The (Capacitated) problem is NP-hard; see Rusmevichientong et al. (2009). So we focus on developing an FPTAS. In the next lemma, shown in Online Appendix I, we give a structural property of an optimal solution to the (Capacitated) problem that is useful to develop an FPTAS.

Lemma 5.1 (Solutions for the Capacitated Problem). *In a nondominated optimal solution $(S_1^*,\ldots,S_m^*)$ to the (Capacitated) problem, for all $k \in \mathcal{M}$, we have $S_k^* \subseteq \{j_k^*+1,\ldots,j_{k+1}^*\}$ for $j_1^*,\ldots,j_{m+1}^*$ with $0 = j_1^* \leq j_2^* \leq \ldots \leq j_m^* \leq j_{m+1}^* = n$.*

This lemma does not immediately yield an efficient algorithm because the optimal assortment S_k^* in stage k may omit products in $\{j_k^*+1,\ldots,j_{k+1}^*\}$. In the (Assortment) problem, by Theorem 3.1, the optimal assortment S_k^* in stage k satisfies $S_k^* = \{j_k^*+1,\ldots,j_{k+1}^*\}$ for j_k^*, j_{k+1}^* with $0 \leq j_k^* \leq j_{k+1}^* \leq n$.

To give an FPTAS for the (Capacitated) problem, we fix a value of $\epsilon \in (0,1)$ and proceed as described in the following two parts.

5.1.1. Part 1: Constructing Candidates.

For each $j,\ell \in \{0,\ldots,n\}$ with $j \leq \ell$, we construct a collection of candidate assortments $\text{Cand}(j,\ell)$ that satisfies the following two properties.

- Correct product interval: For each $\hat{S} \in \text{Cand}(j,\ell)$, we have $\hat{S} \subseteq \{j+1,\ldots,\ell\}$. Thus, a candidate assortment in $\text{Cand}(j,\ell)$ can include only the products in $\{j+1,\ldots,\ell\}$.
- Limited degradation: For each $S \subseteq \{j+1,\ldots,\ell\}$, there exists $\hat{S} \in \text{Cand}(j,\ell)$ such that $W(\hat{S}) \geq (1-\epsilon/4)\,W(S)$, $V(\hat{S}) \leq (1+\epsilon/4)\,V(S)$, and $C(\hat{S}) \leq C(S)$.

Intuitively speaking, noting the objective function of the (Capacitated) problem, we prefer $S \subseteq \mathcal{N}$ with larger $W(S)$, smaller $V(S)$, and smaller $C(S)$. By the second property, for any assortment $S \subseteq \{j+1,\ldots,\ell\}$, there exists a candidate assortment $\hat{S} \in \textsc{Cand}(j, \ell)$ that is almost as preferable. Throughout this section, let $v_{\min} = \min\{v_i : i \in \mathcal{N}\}$, $v_{\max} = \max\{v_i : i \in \mathcal{N}\}$, $w_{\min} = \min\{v_i r_i : i \in \mathcal{N}\}$, and $w_{\max} = \max\{v_i r_i : i \in \mathcal{N}\}$. We construct all collections of candidate assortments $\{\textsc{Cand}(j, \ell) : j, \ell \in \{0, \ldots, n\},\ j \leq \ell\}$ in $O\Big(\frac{n^4}{\epsilon^2} \log\Big(\frac{n w_{\max}}{w_{\min}}\Big) \log\Big(\frac{n v_{\max}}{v_{\min}}\Big)\Big)$ operations. Each collection $\textsc{Cand}(j, \ell)$ includes $O\Big(\frac{n^2}{\epsilon^2} \log\Big(\frac{n w_{\max}}{w_{\min}}\Big) \log\Big(\frac{n v_{\max}}{v_{\min}}\Big)\Big)$ candidate assortments.

5.1.2. Part 2: Combining Candidates.

Having obtained the collection of candidate assortments from part 1, we solve an approximate version of the ($\textsc{Capacitated}$) problem given by

$$\max_{(S_1,\ldots,S_m,j_1,\ldots,j_m)} \left\{ \sum_{k \in \mathcal{M}} \frac{\lambda_k W(S_k)}{\Big(1 + \sum_{q=1}^{k-1} V(S_q)\Big)\Big(1 + \sum_{q=1}^{k} V(S_q)\Big)} : S_k \in \textsc{Cand}(j_k, j_{k+1})\ \ \forall k \in \mathcal{M},\ j_k \leq j_{k+1}\ \ \forall k \in \mathcal{M},\ \sum_{k \in \mathcal{M}} C(S_k) \leq b \right\}, \tag{9}$$

where we follow the convention that $j_{m+1} = n$. Comparing this problem with the ($\textsc{Capacitated}$) problem, we have $S_k \in \textsc{Cand}(j_k, j_{k+1})$. Also, we do not explicitly impose the constraint that $S_k \cap S_q = \varnothing$ for $k \neq q$, but the constraint $S_k \in \textsc{Cand}(j_k, j_{k+1})$ for all $k \in \mathcal{M}$, along with $j_k \leq j_{k+1}$ for all $k \in \mathcal{M}$, ensures that $S_k \cap S_q = \varnothing$ for $k \neq q$. Problem (9) is an approximate version of the ($\textsc{Capacitated}$) problem, in which we can offer only candidate assortments.

Letting $a \vee b = \max\{a, b\}$, we obtain a $(1 - \frac{\epsilon}{4})$-approximate solution to Problem (9) in $O\Big(\frac{n^4 m^3}{\epsilon^4} \log\Big(\frac{n w_{\max}}{w_{\min}}\Big) \log\Big(\frac{n w_{\max}(1 \vee n v_{\max})}{\lambda_m w_{\min}}\Big) \log^2\Big(\frac{n v_{\max}}{v_{\min}}\Big)\Big)$ operations.

In this case, executing the two parts, we get an FPTAS given by the following result.

Theorem 5.1 (FPTAS Under a Space Constraint). *For each $\epsilon \in (0,1)$, we can obtain a $(1-\epsilon)$-approximate solution to the ($\textsc{Capacitated}$) problem in the number of operations*

$$O\left(\frac{n^4 m^3}{\epsilon^4} \log\left(\frac{n w_{\max}}{w_{\min}}\right) \log\left(\frac{n w_{\max}(1 \vee n v_{\max})}{\lambda_m w_{\min}}\right) \log^2\left(\frac{n v_{\max}}{v_{\min}}\right)\right).$$

Proof. We execute the two parts discussed. By the discussion just before the theorem, the number of operations to execute the two parts is given by the expression in the theorem. Let $(S_1^*, \ldots, S_m^*)$ be an optimal solution to the ($\textsc{Capacitated}$) problem. By Lemma 5.1, we know that there exist $0 = j_1^* \leq j_2^* \leq \ldots \leq j_m^* \leq j_{m+1}^* = n$ such that $S_k^* \subseteq \{j_k^* + 1, \ldots, j_{k+1}^*\}$ for all $k \in \mathcal{M}$. After executing part 1, by the second property in part 1, for each $k \in \mathcal{M}$, there exists $\hat{S}_k \in \textsc{Cand}(j_k^*, j_{k+1}^*)$ such that $W(\hat{S}_k) \geq (1 \epsilon/4)\, W(S_k^*)$, $V(\hat{S}_k) \leq (1 + \epsilon/4)\, V(S_k^*)$, and $C(\hat{S}_k) \leq C(S_k^*)$. Because $(S_1^*, \ldots, S_m^*)$ is an optimal solution to the ($\textsc{Capacitated}$) problem, we have $\sum_{k \in \mathcal{M}} C(S_k^*) \leq b$, so noting that $C(\hat{S}_k) \leq C(S_k^*)$ for all $k \in \mathcal{M}$, the solution $(\hat{S}_1, \ldots, \hat{S}_m, j_1^*, \ldots, j_m^*)$ is feasible for Problem (9). After executing part 2, we have a $(1 - \frac{\epsilon}{4})$-approximate solution to Problem (9) as well, which we denote by $(\tilde{S}_1, \ldots, \tilde{S}_m, \tilde{j}_1, \ldots, \tilde{j}_m)$. Therefore, we get

$$\begin{aligned}
&\sum_{k \in \mathcal{M}} \frac{\lambda_k W(\tilde{S}_k)}{\Big(1 + \sum_{q=1}^{k-1} V\big(\tilde{S}_q\big)\Big)\Big(1 + \sum_{q=1}^{k} V\big(\tilde{S}_q\big)\Big)} \\
&\overset{(a)}{\geq} \left(1 - \frac{\epsilon}{4}\right) \sum_{k \in \mathcal{M}} \frac{\lambda_k W\big(\hat{S}_k\big)}{\Big(1 + \sum_{q=1}^{k-1} V\big(\hat{S}_q\big)\Big)\Big(1 + \sum_{q=1}^{k} V\big(\hat{S}_q\big)\Big)} \\
&\overset{(b)}{\geq} \left(1 - \frac{\epsilon}{4}\right) \sum_{k \in \mathcal{M}} \frac{(1 - \frac{\epsilon}{4})\, \lambda_k\, W(S_k^*)}{l\Big(1 + \sum_{q=1}^{k-1} (1 + \frac{\epsilon}{4}) V\big(S_q^*\big)\Big)\Big(1 + \sum_{q=1}^{k} (1 + \frac{\epsilon}{4}) V\big(S_q^*\big)\Big)} \\
&\geq \frac{(1 - \frac{\epsilon}{4})^2}{(1 + \frac{\epsilon}{4})^2} \sum_{k \in \mathcal{M}} \frac{\lambda_k W(S_k^*)}{\Big(1 + \sum_{q=1}^{k-1} V\big(S_q^*\big)\Big)\Big(1 + \sum_{q=1}^{k} V\big(S_q^*\big)\Big)}.
\end{aligned}$$

In this chain of inequalities, (a) holds because $(\tilde{S}_1, \ldots, \tilde{S}_m, \tilde{j}_1, \ldots, \tilde{j}_m)$ is a $(1 - \frac{\epsilon}{4})$-approximate solution to Problem (9), whereas $(\hat{S}_1, \ldots, \hat{S}_m, j_1^*, \ldots, j_m^*)$ is only a feasible solution to Problem (9), whereas (b) holds because $W(\hat{S}_k) \geq (1 - \epsilon/4)\, W(S_k^*)$ and $V(\hat{S}_k) \leq (1 + \epsilon/4)\, V(S_k^*)$. For $\epsilon \in (0,1)$, we have $\frac{(1-\epsilon/4)^2}{(1+\epsilon/4)^2} \geq (1 - \frac{\epsilon}{4})^4 \geq 1 - \epsilon$. In this case, letting z^* be the optimal objective value of the ($\textsc{Capacitated}$) problem and noting that $(S_1^*, \ldots, S_m^*)$ is the optimal solution to the ($\textsc{Capacitated}$) problem, the chain of inequalities presented implies that the solution $(\tilde{S}_1, \ldots, \tilde{S}_m)$ provides an objective value of at least $(1 - \epsilon) z^*$ to the ($\textsc{Capacitated}$) problem.

The number of operations in Theorem 5.1 is polynomial in the input size and $1/\epsilon$, giving an FPTAS. In the next two sections, we discuss how to execute the two parts.

5.2. Part 1: Constructing Collections of Candidate Assortments

We focus on executing part 1. For each $j, \ell \in \{0, \ldots, n\}$ with $j \leq \ell$, we separately construct the collection of candidate assortments $\textsc{Cand}(j, \ell)$. Therefore, we fix j, ℓ throughout this section. Intuitively speaking, to construct the collection of candidate assortments $\textsc{Cand}(j, \ell)$, we use a geometric grid to guess the values of $W(S)$ and $V(S)$ for each possible assortment $S \subseteq \{j + 1, \ldots, \ell\}$. For each guess for the values of $W(S)$ and $V(S)$, we use a dynamic program to find an assortment $\hat{S}$ such that $W(\hat{S})$ and $V(\hat{S})$ are not too far from the guess and the capacity consumption of $\hat{S}$ is as small as possible. The dynamic program that we use is, in spirit, similar to the one that is used for solving the knapsack problem; see, for example, chapter 3 in Williamson and Shmoys (2011) and Desir et al. (2016a). In particular, for fixed $\rho > 0$, we define the geometric grid $\textsc{Dom} = \{(1+\rho)^r : r \in \mathbb{Z}\} \cup \{0\}$. We define the round down operator $\lfloor \cdot \rfloor$ that rounds its argument down to the closest point in $\textsc{Dom}$ when the argument is positive. That is, if $a \geq 0$, then we have $\lfloor a \rfloor = \max\{b \in \textsc{Dom} : b \leq a\}$. If $a < 0$, then we follow the convention that $\lfloor a \rfloor = 0$. Similarly, we define the round up operator $\lceil \cdot \rceil$ that rounds its argument up to the closest point in $\textsc{Dom}$ when the argument is positive. That is, if $a \geq 0$, then we have $\lceil a \rceil = \min\{b \in \textsc{Dom} : b \geq a\}$. If $a < 0$, then we follow the convention that $\lceil a \rceil = -\infty$. For given $(x,y) \in \textsc{Dom}^2$ and (j,ℓ), we consider finding the smallest possible capacity consumption of any assortment $S \subseteq \{j+1,\ldots,\ell\}$ that satisfies $W(S) \geq x$ and $V(S) \leq y$. For this purpose, we use the dynamic program

$$\Theta_i^{\ell}(x,y) = \min_{u_i \in \{0,1\}} \left\{ c_i\, u_i + \Theta_{i+1}^{\ell}(\lfloor x - v_i\, r_i\, u_i \rfloor, \lceil y - v_i\, u_i \rceil) \right\}, \tag{10}$$

where we use the boundary condition that $\Theta_{\ell+1}^{\ell}(x,y) = 0$ if $x \leq 0$ and $y \geq 0$. If, on the other hand, $x > 0$ or $y < 0$, then we have $\Theta_{\ell+1}^{\ell}(x,y) = +\infty$.

In (10), the decision epochs are the products. The action at decision epoch i is whether we offer product i. If we drop the round down and up operators on the right side of (10), then $\Theta_{j+1}^{\ell}(x,y)$ gives the smallest possible capacity consumption of any assortment $S \subseteq \{j+1,\ldots,\ell\}$ that satisfies $W(S) \geq x$ and $V(S) \leq y$. If there is no assortment $S \subseteq \{j+1,\ldots,\ell\}$ such that $W(S) \geq x$ and $V(S) \leq y$, then $\Theta_{j+1}^{\ell}(x,y) = +\infty$. With the round down and up operators on the right side of (10), this dynamic program is only an approximation. Shortly, we put bounds on the two components of the state variable (x,y), in which case, noting that $(x,y) \in \textsc{Dom}^2$, the number of operations required to solve the dynamic program in (10) is a polynomial in the input size.

To construct the collection of candidate assortments $\textsc{Cand}(j,\ell)$, we compute the value functions $\{\Theta_i^{\ell}(x,y) : (x,y) \in \textsc{Dom}^2,\ i = j+1,\ldots,\ell+1\}$ using the dynamic program in (10). Once we compute the value functions, for each $(x,y) \in \textsc{Dom}^2$, starting with state (x,y) and decision epoch $j+1$, we follow the sequence of optimal state-action pairs in the dynamic program in (10). In this way, we obtain an assortment $\hat{S}_{x,y}$ for each $(x,y) \in \textsc{Dom}^2$, which we use as one of the candidate assortments in the collection $\textsc{Cand}(j,\ell)$. Specifically, for each $(x,y) \in \textsc{Dom}^2$, if $\Theta_{j+1}^{\ell}(x,y) < +\infty$, then we construct the assortment $\hat{S}_{x,y}$ using the following algorithm. Throughout this section, we refer to this algorithm as the candidate construction algorithm.

Algorithm 5.1 (Candidate Construction) **Initialization:** Compute the value functions $\{\Theta_i^{\ell}(\cdot,\cdot) : i = j+1,\ldots,\ell+1\}$ using (10). Set $I = j+1$, $\hat{x}_i = x$, and $\hat{y}_i = y$.

Step 1. Set $\hat{u}_i = \operatorname{argmin}_{u_i \in \{0,1\}} \{c_i\, u_i + \Theta_{i+1}^{\ell}(\lfloor \hat{x}_i - v_i\, r_i u_i \rfloor, \lceil \hat{y}_i - v_i\, u_i \rceil)\}$.

Step 2. Set $\hat{x}_{i+1} = \lfloor \hat{x}_i - v_i\, r_i\, \hat{u}_i \rfloor$ and $\hat{y}_{i+1} = \lceil \hat{y}_i - v_i\, \hat{u}_i \rceil$. Increase i by one. If $i < \ell + 1$, then go to step 1; otherwise, stop.

Output: Return $\hat{S}_{x,y} = \{i \in \{j+1,\ldots,\ell\} : \hat{u}_i = 1\}$.

In the next lemma, we show useful properties of the assortment $\hat{S}_{x,y}$ obtained by the algorithm. Specifically, by the next lemma, if there exists an assortment $S \subseteq \{j+1,\ldots,\ell\}$ with $W(S) \geq x$ and $V(S) \leq y$, then we have $\Theta_{j+1}^{\ell}(x,y) < +\infty$, so we execute the algorithm. In this case, considering the assortment $S \subseteq \{j+1,\ldots,\ell\}$ with $W(S) \geq x$ and $V(S) \leq y$, once again, by the next lemma, the output of the candidate construction algorithm $\hat{S}_{x,y}$ satisfies $W(\hat{S}_{x,y}) \geq \frac{1}{(1+\rho)^n}\, x$, $V(\hat{S}_{x,y}) \leq (1+\rho)^n\, y$, and $C(\hat{S}_{x,y}) \leq C(S)$. Recall that we prefer an assortment S with larger $W(S)$, smaller $V(S)$, and smaller $C(S)$. Thus, if ρ is small, then the candidate assortment $\hat{S}_{x,y}$ is almost as preferable as the assortment S.

Lemma 5.2 (Candidate Assortments). *If there exists an assortment $S \subseteq \{j+1,\ldots,\ell\}$ such that $W(S) \geq x$ and $V(S) \leq y$, then we have $\Theta_{j+1}^{\ell}(x,y) < +\infty$, $W(\hat{S}_{x,y}) \geq \frac{1}{(1+\rho)^n}\, x$, $V(\hat{S}_{x,y}) \leq (1+\rho)^n\, y$, and $C(\hat{S}_{x,y}) \leq C(S)$.*

The proof of the lemma is in Online Appendix J. Intuitively speaking, the proof is based on accumulating the errors resulting from the round down and up operators in (10). Next, we discuss how to use the lemma and the candidate construction algorithm to execute part 1. Given $\epsilon \in (0,1)$, we set the accuracy parameter of the geometric grid as $\rho = \frac{1}{8(n+1)}\, \epsilon$. For all $S \neq \emptyset$, we have $W(S) \in [w_{\min}, n\, w_{\max}]$ and $V(S) \in [v_{\min}, n\, v_{\max}]$, so we construct the collection $\textsc{Cand}(j,\ell)$ as

$$\textsc{Cand}(j,\ell) = \Big\{ \hat{S}_{x,y} : (x,y) \in \textsc{Dom}^2,\ x \in [\lfloor w_{\min} \rfloor, \lceil n\, w_{\max} \rceil] \cup \{0\},\ y \in [\lfloor v_{\min} \rfloor, \lfloor n\, v_{\max} \rfloor] \cup \{0\} \Big\}. \tag{11}$$

Noting that $\hat{S}_{x,y} \subseteq \{j+1,\ldots,\ell\}$, we have $\hat{S} \subseteq \{j+1,\ldots,\ell\}$ for all $\hat{S} \in \textsc{Cand}(j,\ell)$. Thus, the collection of candidate assortments $\textsc{Cand}(j,\ell)$ given satisfies the correct product interval property in part 1. It remains to argue that the collection of candidate assortments

$\textsc{Cand}(j,\ell)$ in (11) satisfies the limited degradation property in part 1 as well. In the next lemma, we show that our collection of candidate assortments indeed satisfies this property.

Lemma 5.3 (Limited Degradation). *Considering the collection of candidate assortments* $\textsc{Cand}(j,\ell)$ *in* (11), *for each* $S \subseteq \{j+1,\dots,\ell\}$, *there exists* $\hat{S} \in \textsc{Cand}(j,\ell)$ *such that* $W(\hat{S}) \geq (1-\epsilon/4)\,W(S)$, $V(\hat{S}) \leq (1+\epsilon/4)\,V(S)$, *and* $C(\hat{S}) \leq C(S)$.

Proof. Fix $S \subseteq \{j+1,\dots,\ell\}$ and let $(x,y) \in \textsc{Dom}^2$ be such that $x \in [\lfloor w_{\min}\rfloor, \lceil n\,w_{\max}\rceil] \cup \{0\}$, $y \in [\lfloor v_{\min}\rfloor, \lceil n\,v_{\max}\rceil] \cup \{0\}$, $x \leq W(S) \leq (1+\rho)\,x$, and $y/(1+\rho) \leq V(S) \leq y$. Noting that we have $W(S) \in [w_{\min}, n\,w_{\max}] \cup \{0\}$ and $V(S) \in [v_{\min}, n\,v_{\max}] \cup \{0\}$, there always exists such $(x,y) \in \textsc{Dom}^2$. In this case, because we have $W(S) \geq x$ and $V(S) \leq y$, by Lemma 5.2, the candidate assortment $\hat{S}_{x,y} \in \textsc{Cand}(j,\ell)$ satisfies the inequalities

$$W\left(\hat{S}_{x,y}\right) \geq \frac{1}{(1+\rho)^n}\,x,\;\; V\left(\hat{S}_{x,y}\right) \leq (1+\rho)^n\,y,\; C\left(\hat{S}_{x,y}\right) \leq C(S).$$

Moreover, noting the fact that $W(S) \leq (1+\rho)\,x$ and $V(S) \geq y/(1+\rho)$, the first two preceding inequalities yield $W(\hat{S}_{x,y}) \geq \frac{1}{(1+\rho)^{n+1}}\,W(S)$ and $V(\hat{S}_{x,y}) \leq (1+\rho)^{n+1}\,V(S)$. For all $\delta \in [0,1/2]$ and $n \in \mathbb{Z}_+$, we have the standard inequalities $(1+\delta/n)^n \leq \exp(\delta) \leq 1+2\delta$. Thus, because $\epsilon/8 \leq 1/2$, we get the two chains of inequalities

$$\begin{aligned} W\left(\hat{S}_{x,y}\right) &\geq \frac{1}{(1+\rho)^{n+1}}\,W(S) = \frac{1}{\left(1+\frac{\epsilon}{8\,(n+1)}\right)^{n+1}}\,W(S) \\ &\geq \frac{1}{1+\epsilon/4}\,W(S) \geq (1-\epsilon/4)\,W(S), \\ V\left(\hat{S}_{x,y}\right) &\leq (1+\rho)^{n+1}\,V(S) = \left(1+\frac{\epsilon}{8\,(n+1)}\right)^{n+1} V(S) \\ &\leq (1+\epsilon/4)\,V(S). \end{aligned}$$

Thus, given an assortment $S \subseteq \{j+1,\dots,\ell\}$, there exists $\hat{S}_{x,y} \in \textsc{Cand}(j,\ell)$ such that $W(\hat{S}_{x,y}) \geq (1-\epsilon/4)\,W(S)$, $V(\hat{S}_{x,y}) \leq (1+\epsilon/4)\,V(S)$, and $C(\hat{S}_{x,y}) \leq C(S)$. □

Closing this section, we briefly explain that we can construct all collections of candidate assortments in $O\left(\frac{n^4}{\epsilon^2}\,\log\left(\frac{nw_{\max}}{w_{\min}}\right)\,\log\left(\frac{nv_{\max}}{v_{\min}}\right)\right)$ operations. In particular, to solve the dynamic program in (10), in Online Appendix K, we argue that the smallest nonzero values of x and y in the state variable $(x,y) \in \textsc{Dom}^2$ are, respectively, $\lfloor w_{\min}\rfloor$ and $\lfloor v_{\min}\rfloor$, whereas the largest values of x and y in the state variable $(x,y) \in \textsc{Dom}^2$ are, respectively, $\lceil nw_{\max}\rceil$ and $\lceil nv_{\max}\rceil$. In this case, noting that $\rho = \frac{1}{8\,(n+1)}\epsilon$, the number of state variables that we need to consider is

$$O\left(\frac{\log\left(\frac{nw_{\max}}{w_{\min}}\right)}{\log(1+\rho)} \cdot \frac{\log\left(\frac{nv_{\max}}{v_{\min}}\right)}{\log(1+\rho)}\right) = O\left(\frac{n^2}{\epsilon^2}\log\left(\frac{nw_{\max}}{w_{\min}}\right)\log\left(\frac{nv_{\max}}{v_{\min}}\right)\right). \tag{12}$$

Thus, we can compute $\Theta_i^{\ell}(x,y)$ for all values of the state variable (x,y), $i \in \mathcal{N}$ and $\ell \in \{0,\dots,n\}$ with $i \leq \ell+1$ in $O\left(\frac{n^4}{\epsilon^2}\,\log\left(\frac{nw_{\max}}{w_{\min}}\right)\log\left(\frac{nv_{\max}}{v_{\min}}\right)\right)$ operations. The number of other operations to construct the collections of candidate assortments is negligible, resulting in the desired number of operations to construct all collections. Also, by (11), the collection $\textsc{Cand}(j,\ell)$ includes one assortment for each $(x,y) \in \textsc{Dom}^2$ such that $x \in [\lfloor w_{\min}\rfloor, \lceil n\,w_{\max}\rceil] \cup \{0\}$ and $y \in [\lfloor v_{\min}\rfloor, \lceil n\,v_{\max}\rceil] \cup \{0\}$, so by (12), the collection $\textsc{Cand}(j,\ell)$ includes $O\left(\frac{n^2}{\epsilon^2}\,\log\left(\frac{nw_{\max}}{w_{\min}}\right)\,\log\left(\frac{nv_{\max}}{v_{\min}}\right)\right)$ assortments.

5.3. Part 2: Combining Candidate Assortments

We focus on executing part 2, which obtains an approximate solution to Problem (9). We can solve Problem (9) using dynamic programming. The decision epochs are the stages. At decision epoch k, the action is the candidate assortment S_k offered, whereas the state variable keeps track of j_k such that $S_{k-1} \subseteq \{j_{k-1}+1,\dots,j_k\}$, the accumulated value of $\sum_{q=1}^{k-1} V(S_q)$, and a target expected revenue to generate from the future stages. Thus, we consider the dynamic program

$$\Psi_k(j,u,z) = \min_{\substack{(\ell,S)\,:\,\ell \in \{j,\dots,n\},\\ S \in \textsc{Cand}(j,\ell)}} \left\{C(S) + \Psi_{k+1}\left(\ell\,\lceil u+V(S)\rceil,\,\left\lceil z - \frac{\lambda_k\;W(S)}{(1+u)\,(1+u+V(S))}\right\rceil\right)\right\} \tag{13}$$

with the boundary condition that $\Psi_{m+1}(j,u,z) = 0$ if $z \leq 0$. Otherwise, we have $\Psi_{m+1}(j,u,z) = +\infty$. If we drop the round up operators on the right side of (13), then $\Psi_k(j,u,z)$ gives the smallest total capacity consumption of assortments $(S_k,\dots,S_m)$ such that $S_\ell \in \textsc{Cand}(j_\ell, j_{\ell+1})$ for some $j = j_k \leq j_{k+1} \leq \ldots \leq j_m$, and these assortments provide an expected revenue of at least z in stages $k,\dots,m$ when the assortments $(S_1,\dots,S_{k-1})$ offered in the previous stages satisfy $\sum_{q=1}^{k-1} V(S_q) = u$. In this case, the optimal objective value of Problem (9) is given by $\max\{z \in \mathbb{R} : \Psi_1(0,0,z) \leq b\}$. With the round up operator, the dynamic program in (13) is only an approximation.

To obtain an approximate solution to Problem (9), we use the dynamic program in (13) to compute the value functions $\{\Psi_k(j,u,z) : j = 0,\dots,n,\ (u,z) \in \textsc{Dom}^2,\ k \in \mathcal{M}\}$. Approximating the optimal objective value of Problem (9) as $\hat{z}_{\textsc{App}} = \max\{z \in \textsc{Dom} : \Psi_1(0,0,z) \leq b\}$, we start with the state $(0,0,\hat{z}_{\textsc{App}})$ and follow the optimal state-action pairs in the dynamic program in (13). Specifically, we use the following algorithm to follow the optimal state-action pairs.

Algorithm 5.2 (Candidate Stitching) **Initialization:** Compute the value functions $\{\Psi_k(j,u,z) : j = 0,\ldots,n, (u,z) \in \textsc{Dom}^2, k \in \mathcal{M}\}$ using (13). Set $\hat{z}_{\textsc{App}} = \max\{z \in \textsc{Dom} : \Psi_1(0,0,z) \le b\}$. Initialize $k = 1$, $\hat{j}_k = 0$, $\hat{u}_k = 0$, and $\hat{z}_k = \hat{z}_{\textsc{App}}$.

Step 1. Set

$$\left(\hat{j}_{k+1}, \hat{S}_k\right) = \underset{\substack{(\ell,S)\,:\,\ell \in \{\hat{j}_k,\ldots,n\},\\ S \in \textsc{Cand}(\hat{j}_k,\ell)}}{\arg\min} \left\{ C(S) + \Psi_{k+1}\left(\ell\,,\ \lceil \hat{u}_k + V(S) \rceil\,,\ \left\lceil \hat{z}_k - \frac{\lambda_k W(S)}{(1+\hat{u}_k)(1+\hat{u}_k+V(S))} \right\rceil \right) \right\}$$

Step 2. Set $\hat{u}_{k+1} = \lceil \hat{u}_k + V(\hat{S}_k) \rceil$ and $\hat{z}_{k+1} = \left\lceil \hat{z}_k - \frac{\lambda_k W(\hat{S}_k)}{(1+\hat{u}_k)(1+\hat{u}_k+V(\hat{S}_k))} \right\rceil$. Increase k by one. If $k < m+1$, then go to step 1; otherwise, stop.

Output: Return $(\hat{S}_1,\ldots,\hat{S}_m)$.

Throughout this section, we refer to this algorithm as the candidate stitching algorithm because this algorithm stitches together a solution to Problem (9) using the candidate assortments for different stages. In the next lemma, we show useful properties of the output $(\hat{S}_1,\ldots,\hat{S}_m)$ of the candidate stitching algorithm. In particular, by the next lemma, the output of the candidate stitching algorithm is feasible for Problem (9), satisfying $\sum_{k\in\mathcal{M}} C(\hat{S}_k) \le b$. Furthermore, once again, by the next lemma, using $\textsc{Rev}(S_1,\ldots,S_m)$ to denote the expected revenue from the solution $(S_1,\ldots,S_m)$, $\tilde{z}$ to denote the optimal objective value of Problem (9), and $\hat{z}_{\textsc{App}} = \max\{z \in \textsc{Dom} : \Psi_1(0,0,z) \le b\}$ to denote our approximation of the optimal objective value of Problem (9), we have $\textsc{Rev}(\hat{S}_1,\ldots,\hat{S}_m) \ge \hat{z}_{\textsc{App}} \ge \tilde{z}/(1+\rho)^{3(m+1)}$. Thus, the expected revenue provided by the output of the candidate stitching algorithm is at least as large as our approximation of the optimal objective value of Problem (9). Also, if ρ is small, then our approximation of the optimal objective value of Problem (9) is not too far from the optimal objective value of this problem. The proof of the lemma uses induction over the stages. It is in Online Appendix L.

Lemma 5.4 (Stitching Candidates). *Let $(\hat{S}_1,\ldots,\hat{S}_m)$ be the output of the candidate stitching algorithm, $\tilde{z}$ be the optimal objective of Problem (9), and $\hat{z}_{\textsc{App}} = \max\{z \in \textsc{Dom} : \Psi_1(0,0,z) \le b\}$. We have $\sum_{k\in\mathcal{M}} C(\hat{S}_k) \le b$ and $\textsc{Rev}(\hat{S}_1,\ldots,\hat{S}_m) \ge \hat{z}_{\textsc{App}} \ge \tilde{z}/(1+\rho)^{3m+1}$.*

Next, we discuss how to use the lemma and the candidate stitching algorithm to execute part 2. Given $\epsilon \in (0,1)$, we set the accuracy parameter of the geometric grid as $\rho = \frac{1}{8(3m+1)}\epsilon$. Because $\epsilon/8 \le 1/2$ and $(1+\delta/n)^n \le \exp(\delta) \le 1+2\delta$ for all $\delta \in [0,1/2]$ and $n \in \mathbb{Z}_+$, by Lemma 5.4, we get

$$\textsc{Rev}(\hat{S}_1,\ldots,\hat{S}_m) \ge \frac{1}{(1+\rho)^{3m+1}}\,\tilde{z} = \frac{1}{\left(1+\frac{\epsilon}{8(3m+1)}\right)^{3m+1}}\,\tilde{z} \ge \frac{1}{1+\frac{\epsilon}{4}}\,\tilde{z} \ge \left(1-\frac{\epsilon}{4}\right)\tilde{z},$$

so the output of the candidate stitching algorithm is a $(1-\frac{\epsilon}{4})$-approximate solution to Problem (9) as desired.

Closing this section, we explain that we can execute the candidate stitching algorithm with $\rho = \frac{1}{8(3m+1)}\epsilon$ in $O\left(\frac{n^4m^3}{\epsilon^4}\log\left(\frac{nw_{\max}}{w_{\min}}\right)\log\left(\frac{nw_{\max}(1\vee nv_{\max})}{\lambda_m w_{\min}}\right)\log^2\left(\frac{nv_{\max}}{v_{\min}}\right)\right)$ operations.

To solve the dynamic program in (13), in Online Appendix M, we argue that the largest values of u and z in the state variable $(j,u,z) \in \mathcal{N} \times \textsc{Dom}^2$ are, respectively, $\lceil 2n\,v_{\max} \rceil$ and $\lceil nw_{\max} \rceil$, whereas the smallest nonzero values of u and z in the state variable $(j,u,z) \in \mathcal{N} \times \textsc{Dom}^2$ are, respectively, $\lfloor v_{\min} \rfloor$ and $\lfloor \lambda_m \frac{w_{\min}}{(1+2n\,v_{\max})^2} \rfloor$. Because j in the state variable (j,u,z) takes $O(n)$ possible values and we set $\rho = \frac{1}{8(3m+1)}\epsilon$, the number of state variables we need to consider is

$$O\left(n\,\frac{\log\left(\frac{nv_{\max}}{v_{\min}}\right)}{\log(1+\rho)} \cdot \frac{\log\left(\frac{nw_{\max}}{\lambda_m w_{\min}/(1+2n\,v_{\max})^2}\right)}{\log(1+\rho)}\right) = O\left(\frac{n\,m^2}{\epsilon^2}\log\left(\frac{nv_{\max}}{v_{\min}}\right)\log\left(\frac{nw_{\max}(1\vee n\,v_{\max})}{\lambda_m w_{\min}}\right)\right).$$

In decision epoch k, there are $\sum_{p=\hat{j}^k}^{n} \textsc{Cand}\left(\hat{j}_k, p\right) = O\left(\frac{n^3}{\epsilon^2}\log\left(\frac{nw_{\max}}{w_{\min}}\right)\log\left(\frac{nv_{\max}}{v_{\min}}\right)\right)$ possible actions. Moreover, there are m decision epochs, one for each stage. Therefore, we can solve the dynamic program in (13) in $O\left(\frac{n^4m^3}{\epsilon^4}\log\left(\frac{nw_{\max}}{w_{\min}}\right)\log\left(\frac{nw_{\max}(1\vee nv_{\max})}{\lambda_m w_{\min}}\right)\log^2\left(\frac{nv_{\max}}{v_{\min}}\right)\right)$ operations. The number of other operations to execute the candidate stitching algorithm is negligible, resulting in the desired number of operations to execute the candidate stitching algorithm.

In Online Appendix N, we tailor our FPTAS to the case in which there is a constraint on the total number of offered products and slightly improve its running time. For this case, we also give an exact algorithm with running time polynomial in n but exponential in m.

6. Computational Experiments

We provide three sets of computational experiments. First, we use a data set from Expedia to check the ability of our choice model to predict customer purchases. Second, we test the performance of our approximation algorithm for the joint pricing and assortment-optimization problem. Third, we test the performance of our FPTAS under space constraints. In the second and third sets, we develop upper bounds on the optimal expected revenue and use them to check optimality gaps.

6.1. Prediction Ability on the Data Set from Expedia

We use a data set provided by Expedia as a part of a Kaggle competition; see Kaggle (2013). Our goal is to test the ability of our choice model to predict the purchases of customers.

6.1.1. Experimental Setup. The data set gives the results of search queries for hotels on Expedia. In the data set, the rows correspond to different hotels that are displayed in different search queries. The columns give information on the attributes of the displayed hotel, the results of the search query, and the booking decision of the customer. We preprocess the data set to remove the values that are either missing or uninterpretable as a result of which we end up with 595,965 rows and 15 columns. The first three columns in the data set include the following information. The first column is the unique code for each search query. Using this column, we have access to all of the hotels that are displayed in a particular search query, which is the set of products among which a particular customer makes a choice. The second column is an indicator of whether the customer booked the hotel in the search query. We use this column to identify the purchase of the customer. A customer books at most one hotel in a search query, but it is possible that the customer does not book any hotels. The third column is the display position of the hotel in the search query, which becomes useful when fitting our multinomial logit model with multiple stages. The remaining 12 columns give information on the characteristics of the hotel, such as the star rating, average review score, and displayed price. In Online Appendix O, we explain our approach for preprocessing the data set and give a detailed discussion of the 15 columns that we use.

After processing the data set, the 595,965 rows with which we end up represent 34,561 search queries. The average number of hotels displayed in a search query is 17.24 with the maximum number of hotels being 37. In 83% of the search queries, the customer did not make a booking. To enrich our experimental setup, we use bootstrapping on the data to generate multiple data sets. In each data set, we vary the fraction of the search queries that did not result in a booking. There are a total of 10,000 search queries in each data set that we bootstrap. Using P_0 to denote the fraction of the search queries that did not result in a booking, we sample 10,000 P_0 search queries among the original Expedia search queries that did not result in a booking. Similarly, among the original Expedia search queries that resulted in a booking, we sample $10{,}000\,(1-P_0)$ search queries. Putting these two samples together, we get a data set with 10,000 search queries in which P_0 fraction of them did not result in a booking. For each value of P_0, we repeat the bootstrapping process 50 times to get 50 different data sets. We vary P_0 over $\{0.5, 0.7, 0.9\}$. In this way, we obtain 150 data sets in our computational experiments. The value of P_0 dictates the balance between the customers making and not making a booking in the data set. In our choice model, we capture the preference weight of hotel i in a search query by $v_i = \exp\left(\beta^0 + \sum_{\ell=1}^{12} \beta^\ell x_i^\ell\right)$, where $(x_i^1, \ldots, x_i^{12})$ are the values in the last 12 columns giving the characteristics of hotel i and $(\beta^0, \beta^1, \ldots, \beta^{12})$ are coefficients that we estimate from the data set. Therefore, the parameters of our choice model are the coefficients $(\beta^0, \beta^1, \ldots, \beta^{12})$ and the patience level distribution.

We randomly split each data set into training, validation, and testing data, each of which, respectively, includes 64%, 16%, and 20% of the search queries. The data provides the display position of each hotel in the search query, but fitting our choice model requires having access to the stage in which each hotel is displayed. We proceed under the assumption that each stage corresponds to b hotels in consecutive display positions and choose the best value of b using cross-validation. Specifically, we use the values of $b \in \{1, 3, 5, 10, 20\}$. For each value of b, we use maximum likelihood to fit our choice model to the training data and check the log-likelihood of our fitted choice model on the validation data. We choose the value of b that provides the largest log-likelihood on the validation data. See, for example, the approach used by Vulcano et al. (2012) to fit choice models using maximum likelihood. As a benchmark, we also fit a standard multinomial logit model to the training data. The preference weight of hotel i under this choice model is $v_i = \exp\left(\beta_0 + \sum_{\ell=1}^{12} \beta_\ell x_\ell\right)$. In Online Appendix P, we compare the runtimes for fitting the two choice models. Throughout this section, we refer to our multinomial logit model with impatient customers as IML and the standard multinomial logit model as SML.

6.1.2. Computational Results. We use two performance measures to compare IML and SML. The first measure is the out-of-sample log-likelihood on the testing data. The second is the k-hit score on the testing data. To compute the k-hit score of the fitted IML model, we use $\mathcal{T}$ to denote the set of search queries in the testing data in which the customer made a

Table 1. Comparison of the Fitted IML and SML Models on the Data Set from Expedia

P_0	Out-of-sample log-likelihood				
	IML > SML	IML likelihood	SML likelihood	Average percentage gap	Standard error percentage gap
0.5	50	−3899.65	−3963.64	1.64	0.07
0.7	50	−2701.65	−2766.36	2.40	0.09
0.9	47	−1145.78	−1169.93	2.12	0.18

	One-hit score				Two-hit score				Three-hit score				
P_0	IML > SML	IMLone-hit	Average percentage gap	Standard error, %	IML > SML	IMLtwo-hit	Average percentage gap	Standard percentage gap	IML> SML	IMLthree-hit	Average percentage gap	Standard percentage gap	$\lvert\mathcal{T}\rvert$
0.5	36	0.25	3.81	0.88	37	0.39	3.48	0.61	36	0.50	2.04	0.57	1,003.04
0.7	42	0.24	6.61	0.91	42	0.37	4.24	0.71	32	0.48	3.11	0.68	599.06
0.9	34	0.22	3.04	2.08	35	0.36	5.80	1.57	40	0.47	7.74	1.16	199.84

booking. For each $t \in \mathcal{T}$, we let S_t be the assortment of hotels offered in this search query and i_t be the hotel booked. Using $\phi_i(S)$ to denote the purchase probability of hotel i within assortment S under the fitted IML model, for each t, we let A_t^k be the set of k alternatives with the largest purchase probabilities, which are given by the k largest elements of $\{\phi_i(S_t) : i \in S_t\}$. If $i_t \in A_t^k$, then the hotel booked in search query t has one of the k largest choice probabilities under the fitted IML model. So the k-hit score of the fitted IML model is $\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} 1(i_t \in A_t^k)$. The k-hit score of the fitted SML model is similar. We use $k \in \{1,2,3\}$. For the k-hit score, we focus only on the search queries resulting in a booking because a large fraction of the customers do not book. If we included the search queries not resulting in a booking in the k-hit score, then the k-hit scores would be driven mainly by the customers who do not book, but we want to test our ability to predict the specific hotel booked.

We give our computational results in Table 1. Each row in the table corresponds to a different value of P_0. Recall that we generate 50 data sets for each value of P_0. In the top portion, we compare the out-of-sample log-likelihoods of IML and SML. The first column shows the number of data sets out of 50 in which the out-of-sample log-likelihood of the fitted IML model is larger than that of SML. The second and third columns, respectively, show the average out-of-sample log-likelihood of the fitted IML and SML models, in which the average is over the 50 data sets. The fourth and fifth columns, respectively, show the average and standard error of the percentage gaps between the out-of-sample log-likelihoods of the two fitted choice models, in which the standard error is the standard deviation of the percentage gaps over the 50 data sets divided by $\sqrt{50}$. In the bottom portion, we compare the k-hit scores. The first column shows the number of data sets out of 50 in which the one-hit score of the fitted IML model is larger than that of SML. The second column shows the average one-hit score of the fitted IML model over the 50 data sets. The fourth and fifth columns, respectively, show the average and standard error of the percentage gaps between the one-hit scores of the two fitted choice models. Positive values favor IML. We compare the two- and three-hit scores similarly. Finally, to put the k-hit scores in perspective, we give the average of $|\mathcal{T}|$ over the 50 data sets.

The fitted IML model improves the out-of-sample log-likelihoods of the fitted SML model in 147 out of 150 data sets. To quantify the improvements in the prediction accuracies more clearly, we turn to k-hit scores. The fitted IML model improves the one-hit score of the fitted SML model in 112 out of 150 data sets, providing an average improvement of 4.49%. Noting the three-hit scores, one of the three

Table 2. Estimated Values for the 12 Coefficients in the Fitted IML Model Averaged over the Bootstrapped Data Sets

Attribute	Coefficient
Star rating	0.287
Average review score	0.110
Part of a chain indicator	0.116
Location score	−0.192
Attribute	Coefficient
Accessibility score	0.386
Average historical price	−0.038
Displayed price	−1.202
Promotion indicator	0.127
Attribute	Coefficient
Number of days until stay	−0.111
Number of adults staying	−0.128
Number of children staying	0.065
Saturday stay indicator	−0.051

alternatives with the largest purchase probabilities ends up being the hotel booked by the customer about 50% of the time. The gaps between the k-hit scores are maintained for $k \in \{2, 3\}$, but for large values of k, the k-hit scores for both choice models is naturally one because, as k gets large, the hotel booked by the customer is one of a large number of hotels with a large probability. Our bootstrapped data sets are independent samples. All average gaps in the out-of-sample log-likelihoods and k-hit scores, except for one, are statistically significant in a paired t-test at the 99% level; see chapter 4.6 in Goulden (1939). For $P_0 = 0.9$, the average gap in the one-hit scores is statistically significant at the 90% level.

To get a feel for how the different characteristics of the hotels affect their mean utilities, in Table 2, we provide the estimated values for the coefficients $(\beta^1, \ldots, \beta^{12})$ in the fitted IML model. Recall that the preference weight of hotel i is given by $v_i = \exp\left(\beta^0 + \sum_{\ell=1}^{12} \beta^\ell x_i^\ell\right)$, where $(x_i^1, \ldots, x_i^{12})$ are the values in the last 12 columns giving the characteristics of hotel i, such as the star rating, average review score, and displayed price. Before fitting choice models, we shift and scale the entries in each of the 12 columns of the data set so that the entries in each column have mean zero and variance one, in which case, the entries corresponding to different characteristics of the hotels have roughly the same order of magnitude. The estimated values for the coefficients $(\beta^1, \ldots, \beta^{12})$ are relatively stable from one bootstrapped data set to another, so we provide the average of the estimated parameters over the bootstrapped data sets. Going over some of the values for the coefficients, not surprisingly, larger star rating and larger average review score positively impact the mean utility, whereas larger displayed price negatively impacts the mean utility. Being part of a hotel chain, providing brand familiarity to the customer, positively impacts the mean utility. More interestingly, a larger number of days until the actual day of stay and duration of stay including a Saturday night negatively impact the mean utility. It is reasonable that the customers booking earlier and staying over Saturday night are leisure travelers, so they are more likely to leave without making a booking, resulting in a smaller mean utility. The effects of some characteristics, such as the location and accessibility scores and the numbers of adults and children on the booking, are harder to interpret, but all of the estimated coefficients are statistically significant with p-values less than 10^{-5} when we use the t-test to test the null hypothesis that a coefficient is zero; see chapter 3.1.2 in James et al. (2014).

6.2. Joint Pricing and Assortment Optimization

For joint pricing and assortment optimization, based on Theorem 4.4, we give a simple neighborhood search algorithm with 87.8% performance guarantee and test its performance.

6.2.1. Experimental Setup. In our neighborhood search algorithm, we start with a solution to the (Pricing-Assortment) problem that offers all products in the first stage and charges the corresponding optimal prices for the products. By Theorem 4.4, this solution provides at least 87.8% of the optimal expected revenue. Given the current sequence of assortments that we offer, we check all neighbors of the current sequence of assortments, using an appropriately defined neighborhood. For each sequence of assortments in the neighborhood, we compute the corresponding optimal prices to charge. Among all sequences of assortments in the neighborhood and their corresponding prices, we pick the one that provides the largest expected revenue. Recall that we can use the approach in Section 4.1 to compute the optimal prices to charge for a given sequence of assortments. We update the current sequence of assortments to be this best sequence in the neighborhood and repeat the process starting from the updated current sequence of assortments. If no sequence of assortments in the neighborhood, along with the corresponding optimal prices, improves the expected revenue from the current sequence of assortments, then we stop. In our neighborhood search algorithm, we define the neighborhood of a sequence of assortments as all sequences of assortments obtained by moving one product from one stage to another, so the neighborhood of the sequence of assortments $(S_1, \ldots, S_m)$ is $\{(S_1, \ldots, S_k \setminus \{i\}, \ldots, S_\ell \cup \{i\}, \ldots, S_m) : \forall i \in S_k,\ k, \ell \in \mathcal{M},\ k \neq \ell\}$. To complement the neighborhood search algorithm, in Online Appendix Q, we also give an efficiently computable upper

Table 3. Performance of the Neighborhood Search Algorithm for Joint Pricing and Assortment Optimization

Parameter configuration (C,σ,a)	$m=6$ Average gap, %	Maximum gap, %	Standard deviation gap, %	Average improvement, %	$m=8$ Average gap, %	Maximum gap, %	Standard deviation gap, %	Average improvement, %	$m=10$ Average gap, %	Maximum gap, %	Standard deviation gap, %	Average improvement, %
(3, 0.5, +∞)	1.38	1.59	0.09	4.98	1.95	2.17	0.09	4.99	2.52	2.74	0.09	4.99
(3, 0.5, 0.5)	1.15	1.17	0.01	2.73	1.63	1.66	0.01	3.70	2.12	2.22	0.03	4.28
(3, 0.5, 0.0)	0.78	0.79	0.00	0.55	1.04	1.10	0.01	1.01	1.29	1.38	0.02	1.36
(3, 0.5, −0.1)	0.70	0.70	0.00	0.24	0.89	0.90	0.01	0.50	1.06	1.07	0.00	0.71
(3, 1.0, +∞)	1.53	2.61	0.30	4.82	2.10	3.18	0.31	4.83	2.67	3.75	0.31	4.83
(3, 1.0, 0.5)	1.18	1.57	0.08	2.70	1.66	2.12	0.12	3.66	2.20	2.86	0.17	4.21
(3, 1.0, 0.0)	0.78	0.90	0.02	0.55	1.03	1.04	0.00	1.01	1.29	1.38	0.02	1.36
(3, 1.0, −0.1)	0.70	0.70	0.00	0.24	0.89	0.92	0.01	0.50	1.06	1.14	0.02	0.70
(5, 0.5, +∞)	1.49	1.77	0.16	4.87	2.06	2.34	0.17	4.88	2.63	2.92	0.17	4.88
(5, 0.5, 0.5)	1.16	1.18	0.01	2.73	1.64	1.72	0.03	3.69	2.15	2.27	0.05	4.25
(5, 0.5, 0.0)	0.78	0.79	0.00	0.55	1.04	1.09	0.02	1.00	1.29	1.33	0.01	1.36
(5, 0.5, −0.1)	0.70	0.71	0.00	0.24	0.89	0.91	0.01	0.50	1.06	1.11	0.02	0.70
(5, 1.0, +∞)	1.84	2.95	0.46	4.49	2.41	3.52	0.46	4.50	2.98	4.09	0.46	4.50
(5, 1.0, 0.5)	1.21	2.28	0.22	2.67	1.78	2.59	0.24	3.54	2.36	3.25	0.31	4.04
(5, 1.0, 0.0)	0.78	0.86	0.02	0.55	1.07	1.21	0.06	0.97	1.31	1.46	0.05	1.34
(5, 1.0, −0.1)	0.70	0.75	0.01	0.23	0.88	0.89	0.00	0.50	1.07	1.40	0.07	0.69

bound on the optimal expected revenue in the (Pricing-Assortment) problem. In our computational experiments, we randomly generate a large number of test problems and compare the expected revenue from the solution obtained by our neighborhood search algorithm with the upper bound on the optimal expected revenue.

In all of our test problems, the number of products is $n=20$ and the price sensitivity is $\beta=1$. Working with other values for the price sensitivity is equivalent to scaling the prices of the products with the same constant. We use the following approach to come up with the parameters $\{\alpha_i : i \in \mathcal{N}\}$. We have C product clusters. We randomly assign each product to a cluster. If products i and j are in the same cluster, then the values of α_i and α_j are close. Specifically, cluster c has the centroid γ_c. We set the centroid of cluster c as $\gamma_c = c-0.5$ for all $c=1,\dots,C$. If product i belongs to cluster c, then we generate κ_i from the normal distribution with mean γ_c and standard deviation σ, where σ is a parameter that we vary. We set $\alpha_i = \kappa_i - \Delta$, where we have $\Delta = \log\sum_{i\in\mathcal{N}} e^{\kappa_i} - \log 9$. In this case, if we offer all products in the first stage and charge a price of zero for them, then a customer leaves without a purchase with probability 0.1. Using the random variable Y to capture the patience level of a customer, the probability mass function of Y is given by $\mathbb{P}\{Y=k\} = e^{a\cdot k}/\sum_{\ell\in\mathcal{M}} e^{a\cdot\ell}$, where a is another parameter that we vary. Negative and positive values for a yield, respectively, left- and right-skewed distributions. If $a=+\infty$, then $Y=m$ with probability one, so the customers are willing to wait until the last stage.

Recalling that m is the number of stages, varying $m \in \{6,8,10\}$, $C \in \{3,5\}$, $\sigma \in \{0.5,1.0\}$, and $a \in \{+\infty, 0.5, 0.0, -0.1\}$, we obtain 48 parameter configurations. In each parameter configuration, we generate 25 problem instances using the approach in the previous paragraph.

6.2.2. Computational Results. We show our computational results in Table 3. In this table, the first column shows the parameter configuration using the tuple (C,σ,a), where C, σ, and a are as discussed in our experimental setup. In the rest of the table, there are three blocks, each with four columns. Each block corresponds to a particular value for the parameter m. In each block, the first column shows the average percentage gap between the upper bound on the optimal expected revenue and the expected revenue from the solution obtained by our neighborhood search algorithm, in which the average is computed over the 25 problem instances in a parameter configuration. Specifically, using Rev^k and UB^k to denote, respectively, the expected revenue from the solution obtained by the neighborhood search algorithm and the upper bound on the optimal expected revenue for problem instance k, the first column shows the average of the data $\left\{100 \cdot \frac{\mathrm{UB}^k - \mathrm{Rev}^k}{\mathrm{UB}^k} : k = 1, \ldots, 25\right\}$. The second and third columns, respectively, show the maximum and standard deviation of the same data. The fourth column shows the average percentage gap between the expected revenues from the initial and final solutions in the neighborhood search algorithm, capturing the improvement provided by this algorithm over using the solution that offers all products in the first stage.

Our results indicate that our neighborhood search algorithm performs quite well. Over all of our test problems, the average gap between the upper bound on the optimal expected revenue and the expected revenue from the neighborhood search algorithm is 1.43%. The gaps tend to increase as the number of stages gets larger. Without knowing the optimal expected revenue, it is difficult to tell whether the increase in the gaps is due to a degradation in the upper bounds or a degradation in the expected revenues from the neighborhood search algorithm. However, the upper bound that we give in Online Appendix Q is based on treating $\left(\sum_{i \in S_1} e^{\alpha_i}, \ldots, \sum_{i \in S_m} e^{\alpha_i}\right)$ in the expected revenue expression in (5) as continuous quantities whose sum does not exceed $\sum_{i \in \mathcal{N}} e^{\alpha_i}$. Intuitively speaking, this assumption becomes harder to justify when the number of stages gets larger. Overall, the performance of the neighborhood search algorithm is better than its 87.8% theoretical performance guarantee. The improvement provided by the neighborhood search algorithm over the initial solution that offers all products in the first stage can get as large as 4.99%. The improvements are most visible for problem instances with $a \in \{+\infty, 0.5\}$. For these problem instances, the patience level distribution has a significant mass for larger patience levels, so the customers tend to have larger patience levels, in which case, it becomes more important to explore solutions that offer assortments in the later stages. On the other hand, for problem instances with $a \in \{0.0, -0,1\}$, the patience level distribution has a significant mass for smaller patience levels. In this case, the customers tend to have smaller patience levels, so focusing on solutions that offer assortments only in a few earlier stages appears to be adequate to get good solutions. For problem instances with $a = +\infty$, $a = 0.5$, $a = 0.0$, and $a = -0.1$, the average number of neighbors that the neighborhood search algorithm visits before termination are, respectively, 38.31, 9.02, 1.89, and 1.74, which also indicates that as the value of a gets larger so that the customers tend stay in the system for a larger number of stages before they run of patience, it becomes more important to explore solutions that offer assortments in later stages. The runtime for the neighborhood search algorithm ranges from 0.11 and 5.14 seconds with larger runtimes corresponding to problem instances with larger values for m and a.

6.3. Assortment Optimization Under a Space Constraint

We test the practical performance of the FPTAS that we give in Section 5 for the assortment optimization problem under a space constraint.

6.3.1. Experimental Setup. To assess the optimality gaps for the solutions obtained by our FPTAS, in Online Appendix R, we give an efficiently computable upper bound on the optimal objective value of the (Capacitated) problem. In our computational experiments, we randomly generate a large number of test problems and compare the expected revenue from the solution obtained by our FPTAS with the upper bound on the optimal expected revenue. We use the following approach to generate our test problems. In all of our test problems, the number of products is $n = 20$. To come up with the revenue r_i of product i, we generate r_i from the uniform distribution over $[1, 10]$. We reindex $(r_1, \ldots, r_n)$ so that $r_1 \geq r_2 \geq \ldots \geq r_n$. To come up with the preference weight v_i of product i, we generate γ_i from the uniform distribution $[1, 10]$ and set $v_i = \gamma_i / \Delta$, where we have $\Delta = P_0 \sum_{i \in \mathcal{N}} \gamma_i / (1 - P_0)$ and P_0 is a parameter that we vary. In this case, if we offer all products in the first stage, then a customer leaves without a purchase with probability P_0. After generating the preference weights, we process them to come up with two problem classes for the preference weights. In the first problem class, we leave the preference weights untouched. In the second problem class, we reindex $(v_1, \ldots, v_n)$ so that $v_1 \leq v_2 \leq \ldots \leq v_n$. Thus, recalling that $r_1 \geq r_2 \geq \ldots \geq r_n$, in the second problem class, the products with larger revenues have smaller preference weights, so the more expensive products are less attractive. We refer to the first and second problem classes, respectively, as "U" for unordered and "O" for ordered. We use the same approach that we use in Section 6.2 for the joint pricing and assortment-optimization problem to come up with the distribution for the patience levels. Recall that the parameter a controls the skewness of the distribution of the patience levels. In all of our test problems, to come up with the space consumptions $\{c_i : i \in \mathcal{N}\}$ and the space availability b, we generate c_i from the uniform distribution over $[0, 1]$ and set $b = 5$.

Using T to denote the problem class for the preference weights, varying $m \in \{6, 8, 10\}$, $P_0 \in \{0.1, 0, 3\}$, $T \in \{\mathrm{U}, \mathrm{O}\}$, and $a \in \{+\infty, 0.5, 0.0, -0.1\}$, we obtain 48 parameter configurations. We generate 25 problem instances in each parameter configuration.

Table 4. Performance of the FPTAS for Assortment Optimization Under a Space Constraint

Parameter configuration (P_0, T, a)	$m=6$ Average gap, %	Maximum gap, %	Standard deviation gap, %	$m=8$ Average gap, %	Maximum gap, %	Standard deviation gap, %	$m=10$ Average gap, %	Maximum gap, %	Standard deviation gap, %
(0.1, U, $+\infty$)	2.92	3.88	0.56	3.18	4.20	0.56	3.27	4.31	0.58
(0.1, U, 0.5)	2.40	3.22	0.42	2.53	3.44	0.56	2.79	3.71	0.47
(0.1, U, 0.0)	2.17	3.09	0.49	2.28	3.71	0.51	2.19	3.73	0.65
(0.1, U, −0.1)	2.12	3.09	0.55	2.09	3.07	0.50	2.11	3.21	0.56
(0.1, O, $+\infty$)	2.77	4.00	0.60	3.11	4.27	0.60	3.30	4.47	0.59
(0.1, O, 0.5)	2.14	2.77	0.40	2.38	3.72	0.60	2.72	4.00	0.57
(0.1, O, 0.0)	1.96	3.32	0.62	2.15	3.34	0.61	1.99	3.81	0.53
(0.1, O, −0.1)	2.25	3.54	0.53	2.01	2.93	0.54	2.03	3.04	0.54
(0.3, U, $+\infty$)	2.52	3.58	0.45	2.80	3.84	0.43	2.93	4.08	0.46
(0.3, U, 0.5)	2.15	3.22	0.52	2.25	3.01	0.45	2.16	3.03	0.47
(0.3, U, 0.0)	1.87	2.71	0.40	1.78	2.59	0.45	1.87	2.86	0.49
(0.3, U, −0.1)	1.80	2.65	0.43	1.79	2.71	0.42	2.02	2.94	0.45
(0.3, O, $+\infty$)	2.37	3.20	0.47	2.69	3.58	0.46	2.85	3.85	0.46
(0.3, O, 0.5)	1.74	2.67	0.50	1.96	3.19	0.52	2.07	2.99	0.43
(0.3, O, 0.0)	1.59	2.73	0.55	1.52	2.48	0.46	1.73	2.63	0.53
(0.3, O, −0.1)	1.50	2.52	0.53	1.54	2.47	0.48	1.55	2.75	0.52

6.3.2. Computational Results. We executed our FPTAS with $\epsilon = 1/2$ to obtain a $\frac{1}{2}$-approximate solution to the (Capacitated) problem. Even with this setting, our FPTAS obtains solutions with expected revenues within 5% of the upper bound on the optimal expected revenue. The large number of test problems in our experimental setup prevents us from reporting results for theoretical performance guarantees better than 50%, but a limited number of runs indicates that, if we use $\epsilon = \frac{1}{4}$, then we decrease the percentage gap between the upper bound and the performance of our FPTAS by about 1%. We show our computational results in Table 4. The layout of this table is similar to that of Table 3. In the first column, we use the tuple (P_0, T, a) to show the parameter configuration. The rest of the table has three blocks of three columns. Each block corresponds to a particular value for the parameter m. In each block, the three columns, respectively, show the average, maximum, and standard deviation for the percentage gap between the upper bound on the optimal expected revenue and the expected revenue from the solution obtained by our FPTAS, in which the average, maximum, and standard deviation are computed over the 25 problem instances in a parameter configuration. Over all of our test problems, the average gap between the upper bound and the expected revenue from our FPTAS is 2.25% and the maximum gap is 4.47%. The gaps increase only slightly as the number of stages gets larger. Overall, the performance of our FPTAS is substantially stronger than its theoretical performance guarantee of 50%.

The runtime for our FPTAS ranges from 26.23 to 36.12 minutes. Note that the size of our test problems makes full enumeration impossible because the number of possible sequences of assortments is $O(m^n)$. Considering the candidate construction and candidate stitching algorithms in Sections 5.2 and 5.3, a major portion of the runtime is spent for candidate construction, particularly because many of the candidate assortments that we construct end up being duplicates of each other, resulting in substantial savings in the runtime for candidate stitching. The runtime for the candidate stitching algorithm ranges from 2.08 to 18.09 seconds, and the larger runtimes correspond to the test problems with larger number of stages. The remaining portion of the runtime is for the candidate construction algorithm. In Table 5, we give the runtime in minutes for our FPTAS for the values of $\epsilon \in \{3/4, 1/2, 1/4, 1/8\}$, averaged over four representative problem instances with $m = 10$ stages. We use four problem instances as the runtime with $\epsilon = 1/8$ is a few hours. In our computational experiments, the runtime of our FPTAS increases roughly quadratically with $\frac{1}{\epsilon}$ because many of the candidate assortments that we construct, as mentioned earlier in this paragraph, end up being duplicates of each other.

Table 5. Runtime for the FPTAS for Different Values of ϵ

ϵ	3/4	1/2	1/4	1/8
Runtime, minutes	14.37	31.05	118.43	464.27

7. Conclusions

Our work in this paper opens up several research directions to pursue. We can explore efficient solution methods for the assortment-optimization problem when there is a constraint on the space consumption or the number of products offered in each stage. An analogue of Lemma 5.1 does not hold under a constraint on the space consumption or the number of products offered in each stage. That is, we can have a pair of products such that it is optimal to offer the one with the smaller revenue in an earlier stage and the one with the larger revenue in a later stage. For example, consider a problem instance with three products and two stages. The revenues and preference weights of the products are $r_1 = 10$, $r_2 = 7$, $r_3 = 6$, $v_1 = 0.4$, $v_2 = 0.9$, and $v_3 = 0.3$. The distribution of the patience levels is given by $\lambda_1 = \lambda_2 = 1$. If we can offer at most two products in the first stage and at most one product in the second stage, then the unique optimal assortments in the first and second stages are, respectively, $\{1,3\}$ and $\{2\}$, offering product 3 with revenue 6 in the first stage and product 2 with revenue 7 in the second stage. Similarly, we have not been able to characterize structural properties of a near-optimal sequence of assortments to offer in the joint pricing and assortment optimization problem when there is a constraint on the number of products offered in each stage. Thus, the joint pricing and assortment-optimization problem under a constraint on the number of products offered in each stage is an open problem. Considering another research direction, in Online Appendix D, we give a dynamic program to find a solution for the (Pricing) problem with an additive performance guarantee. It would be useful to extend this work to find a solution with a multiplicative performance guarantee without using convex optimization tools.

Also, the running time of our FPTAS is $O\left(\frac{n^4 m^3}{\epsilon^4}\log\left(\frac{nw_{\max}}{w_{\min}}\right)\log\left(\frac{nw_{\max}(1\vee nv_{\max})}{\lambda_m w_{\min}}\right)\log^2\left(\frac{nv_{\max}}{v_{\min}}\right)\right)$, depending on the parameters $w_{\max}$, $w_{\min}$, $v_{\max}$, $v_{\min}$, and λ_m. We can work on removing the dependence on these parameters to obtain a strongly polynomial running time. We can remove these dependencies partially. In particular, when solving the dynamic program in (10), we can guess the largest value of $v_i r_i$ for an offered product i and the largest value of v_j for an offered product j. Letting $\hat{w}$ and $\hat{v}$ be these two guesses, we can argue that the largest values of x and y in the state variable $(x,y) \in \textsc{Dom}^2$ would, respectively, be $\lceil n\hat{w}\rceil$ and $\lceil n\hat{v}\rceil$, whereas the smallest nonzero values of x and y in the state variable $(x,y) \in \textsc{Dom}^2$ would, respectively, be $\lfloor \hat{w}\rfloor$ and $\lfloor \hat{v}\rfloor$. Thus, noting that there are n^2 possible guesses for $(\hat{w},\hat{v})$, the number of candidate assortments in the collection $\textsc{Cand}(j,\ell)$ would be $O\left(\frac{n^4}{\epsilon^2}(\log n)^2\right)$, which is strongly polynomial, but it is not necessarily smaller than the number of candidate assortments $O\left(\frac{n^2}{\epsilon^2}\log\left(\frac{nw_{\max}}{w_{\min}}\right)\log\left(\frac{nv_{\max}}{v_{\min}}\right)\right)$ discussed at the end of Section 5.2. In this case, following the same line of analysis in Section 5.3, the running time of our FPTAS would be $O\left(\frac{n^6 m^3}{\epsilon^4}(\log n)^2\log\left(\frac{nv_{\max}}{v_{\min}}\right)\log\left(\frac{nw_{\max}(1\vee n\, v_{\max})}{\lambda_m w_{\min}}\right)\right)$. We can use a similar idea for the dynamic program in (13) to deal with u in the state variable $(j,u,z) \in \mathcal{N} \times \textsc{Dom}^2$, yielding the running time $O\left(\frac{n^7 m^3}{\epsilon^4}(\log n)^3\log\left(\frac{nw_{\max}(1\vee n\, v_{\max})}{\lambda_m w_{\min}}\right)\right)$ for our FPTAS, but dealing with z in the state variable $(j,u,z) \in \mathcal{N} \times \textsc{Dom}^2$ appears to be difficult because of the term λ_k in (13).

Finally, in Online Appendix H, we show that the joint pricing and assortment-optimization problem is NP-hard when the prices take values over a discrete set. We do not know the computational complexity of the problem when the prices take values over a continuum. Similarly, in Online Appendix N, considering the case in which there is a constraint on the total number of offered products, we give an algorithm to find the optimal sequence of assortments to offer, but the running time of this algorithm increases exponentially with the number of stages. We do not know the complexity of the problem when the number of stages is also an input.

Acknowledgments

The authors thank the associate editor and two anonymous referees whose comments substantially improved both the technical results and computational experiments. The current proof of Theorem 4.1 was suggested by one of the referees, simplifying our earlier proof. The authors thank Shane Henderson for feedback on a presentation of an earlier version of the work. PG and YM were graduate students at the time of this work. They are listed alphabetically, and so are the remaining authors. The work in this paper is independent discovery by two different sets of authors: PG, NC, GG, and AL and YM, PR, and HT.

References

Aouad A, Segev D (2018) Display optimization for vertically differentiated locations under multinomial logit preferences. Technical report, Massachusetts Institute of Technology, Cambridge, MA.

Aouad A, Farias V, Levi R (2016) Assortment optimization under consider-then-choose choice models. Technical report, Massachusetts Institute of Technology, Cambridge, MA.

Aouad A, Feldman J, Segev D, Zhang DJ (2019) Click-based MNL: Algorithmic frameworks for modeling click data in assortment optimization. Technical report, Washington University, St. Louis, MO.

Blanchet J, Gallego G, Goyal V (2016) A Markov chain approximation to choice modeling. *Oper. Res.* 64(4):886–905.

Boyd S, Vandenberghe L (2004) *Convex Optimization* (Cambridge University Press, Cambridge, UK).

Bront JJM, Mendez Diaz I, Vulcano G (2009) A column generation algorithm for choice-based network revenue management. *Oper. Res.* 57(3):769–784.

Davis JM, Gallego G, Topaloglu H (2014) Assortment optimization under variants of the nested logit model. *Oper. Res.* 62(2):250–273.

Derakhshan M, Golrezaei N, Manhshadi V, Mirrokni V (2018) Product ranking on online platforms. Technical report, Massachusetts Institute of Technology, Cambridge, MA.

Desir A, Goyal V, Zhang J (2016a) Near-optimal algorithms for capacity constrained assortment optimization. Technical report, Columbia University, New York.

Desir A, Goyal V, Jagabathula S, Segev D (2016b) *Assortment Optimization Under the Mallows Model.* Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R, eds. Advances in Neural Information Processing Systems, vol. 29 (Curran Associates, Inc.), 4700–4708.

Farias VF, Jagabathula S, Shah D (2013) A non-parametric approach to modeling choice with limited data. *Management Sci.* 59(2):305–322.

Feldman J, Segev D (2019) Improved approximation schemes for MNL-driven sequential assortment optimization. Technical report, Washington University, St. Louis, MO.

Feldman J, Paul A, Topaloglu H (2019) Assortment optimization with small consideration sets. *Oper. Res.* 67(5):1283–1299.

Flores A, Berbeglia G, van Hentenryck P (2019) Assortment optimization under the sequential multinomial logit model. *Eur. J. Oper. Res.* 273(3):1052–1064.

Gallego G, Wang R (2014) Multi-product price optimization and competition under the nested attraction model. *Oper. Res.* 62(2):450–461.

Gallego G, Iyengar G, Phillips R, Dubey A (2004) Managing flexible products on a network. CORC Technical Report TR-2004-01.

Gallego G, Li A, Truong VA, Wang X (2021) Approximation algorithms for product framing and pricing. *Oper. Res.* Forthcoming.

Goulden CH (1939) *Methods of Statistical Analysis* (John Wiley & Sons, New York).

Hopp WJ, Xu X (2005) Product line selection and pricing with modularity in design. *Manufacturing Service Oper. Management* 7(3):172–187.

Jagabathula S (2016) Assortment optimization under general choice. Technical report, New York University.

James G, Witten D, Hastie T, Tibshirani R (2014) *An Introduction to Statistical Learning* (Springer, New York).

Kaggle (2013) Personalize Expedia hotel searches. Accessed August 5, 2019, https://www.kaggle.com/c/expedia-personalized-sort.

Li H, Huh WT (2011) Pricing multiple products with the multinomial logit and nested models: Concavity and implications. *Manufacturing Service Oper. Management* 13(4):549–563.

Li H, Webster S (2017) Optimal pricing of correlated product options under the paired combinatorial logit model. *Oper. Res.* 65(5):1215–1230.

Liu N, Ma Y, Topaloglu H (2021) Assortment optimization under the multinomial logit model with sequential offerings. *INFORMS J. Comput.* Forthcoming.

Mendez-Diaz I, Bront JJM, Vulcano G, Zabala P (2014) A branch-and-cut algorithm for the latent-class logit assortment problem. *Discrete Appl. Math.* 164(1):246–263.

Rusmevichientong P, Shen Z-JM, Shmoys DB (2009) A PTAS for capacitated sum-of-ratios optimization. *Oper. Res. Lett.* 37(4):230–238.

Rusmevichientong P, Shen Z-JM, Shmoys DB (2010) Dynamic assortment optimization with a multinomial logit choice model and capacity constraint. *Oper. Res.* 58(6):1666–1680.

Rusmevichientong P, Shmoys D, Tong C, Topaloglu H (2014) Assortment optimization under the multinomial logit model with random choice parameters. *Production Oper. Management* 23(11):2023–2039.

Song J-S, Xue Z (2007) Demand management and inventory control for substitutable products. Technical report, Duke University, Durham, NC.

Sumida M, Gallego G, Rusmevichientong P, Topaloglu H, Davis JM (2019) Revenue-utility tradeoff in assortment optimization under the multinomial logit model with totally unimodular constraints. Technical report, Cornell University, Ithaca, NY.

Talluri K, van Ryzin GJ (2004) Revenue management under a general discrete choice model of consumer behavior. *Management Sci.* 50(1):15–33.

Talluri KT, van Ryzin GJ (2005) *The Theory and Practice of Revenue Management* (Kluwer Academic Publishers, Boston).

Vulcano G, van Ryzin GJ, Ratliff R (2012) Estimating primary demand for substitutable products from sales transaction data. *Oper. Res.* 60(2):313–334.

Wang R (2012) Capacitated assortment and price optimization under the multinomial logit model. *Oper. Res. Lett.* 40(6):492–497.

Wang R, Sahin O (2018) The impact of consumer search cost on assortment planning and pricing. *Management Sci.* 64(8):3649–3666.

Williamson DP, Shmoys DB (2011) *The Design of Approximation Algorithms* (Cambridge University Press).

Zhang D, Lu Z (2013) Assessing the value of dynamic pricing in network revenue management. *INFORMS J. Comput.* 25(1):102–115.

Zhang H, Rusmevichientong P, Topaloglu H (2018) Technical note: Multiproduct pricing under the generalized extreme value models with homogeneous price sensitivity parameters. *Oper. Res.* 66(6):1559–1570.

Pin Gao is an assistant professor in the School of Data Science, Chinese University of Hong Kong, Shenzhen. His research interests include revenue management, innovation, algorithm design, and mechanism design.

Yuhang Ma obtained her PhD from the School of Operations Research and Information Engineering at Cornell University and now works at Aurora Innovation, Inc. Her research interests include optimization algorithm design and approximate dynamic programming with applications in revenue management and autonomous driving.

Ningyuan Chen is currently an assistant professor in the Department of Management at the University of Toronto, Mississauga, and cross-appointed at the Rotman School of Management, University of Toronto. He is interested in various approaches to making data-driven decisions in the context of revenue management, including online learning and machine learning. His research is also broadly related to statistics.

Guillermo Gallego is the department head of Industrial Engineering and Decision Analytics and the Crown Worldwide Professor of Engineering at the HKUST. He was named a MSOM Fellow (2013) and an INFORMS Fellow (2012). His awards include the INFORMS Impact Prize (2016), the Management Science Best Paper (2017), and the Best OM paper in Operations Research (2019). His research interests are dynamic pricing, discrete choice models, assortment optimization, pricing analytics and dynamic programming.

Anran Li is an assistant professor of operations management in the Department of Management at the London School of Economics. She is interested in developing data-

driven algorithms, especially with applications to revenue management and e-commerce. Her recent research has focused on behavioral consumer choice modeling and online personalized assortment optimization, pricing and products ranking.

Paat Rusmevichientong is a professor of data sciences and operations at the University of Southern California Marshall School of Business. His research interests focus on revenue management, choice modeling, pricing, assortment optimization, and large-scale dynamic programming.

Huseyin Topaloglu is a professor in the School of Operations Research and Information Engineering at Cornell University. His research focuses on revenue management, assortment optimization, transportation logistics, and approximate dynamic programming.

CORRECTION

In this article, "Assortment Optimization and Pricing Under the Multinomial Logit Model with Impatient Customers: Sequential Recommendation and Selection" by Pin Gao, Yuhang Ma, Ningyuan Chen, Guillermo Gallego, Anran Li, Paat Rusmevichientong, and Huseyin Topaloglu (*Operations Research* vol. 69, no. 5, pp. 1509–1532), Guillermo Gallego's affiliation has been corrected.