Manufacturing & Service Operations Management

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Computing Virtual Nesting Controls for Network Revenue Management Under Customer Choice Behavior

Garrett van Ryzin, Gustavo Vulcano,

To cite this article:
Garrett van Ryzin, Gustavo Vulcano, (2008) Computing Virtual Nesting Controls for Network Revenue Management Under Customer Choice Behavior. Manufacturing & Service Operations Management 10(3):448-467. https://doi.org/10.1287/msom.1070.0210

M&SOM MANUFACTURING & SERVICE OPERATIONS MANAGEMENT
Vol. 10, No. 3, Summer 2008, pp. 448–467
ISSN 1523-4614 | EISSN 1526-5498 | 08 | 1003 | 0448

informs

DOI 10.1287/msom.1070.0210

Computing Virtual Nesting Controls for Network Revenue Management Under Customer Choice Behavior

Garrett van Ryzin
Graduate School of Business, Columbia University, New York, New York 10027,
gjv1@columbia.edu

Gustavo Vulcano
Stern School of Business, New York University, New York, New York 10012,
gvulcano@stern.nyu.edu

We consider a revenue management, network capacity control problem in a setting where heterogeneous customers choose among the various products offered by a firm (e.g., different flight times, fare classes, and/or routings). Customers may therefore substitute if their preferred products are not offered. These individual customer choice decisions are modeled as a very general stochastic sequence of customers, each of whom has an ordered list of preferences. Minimal assumptions are made about the statistical properties of this demand sequence. We assume that the firm controls the availability of products using a virtual nesting control strategy and would like to optimize the protection levels for its virtual classes accounting for the (potentially quite complex) choice behavior of its customers.

We formulate a continuous demand and capacity approximation for this problem, which allows for the partial acceptance of requests for products. The model admits an efficient calculation of the sample path gradient of the network revenue function. This gradient is then used to construct a stochastic steepest ascent algorithm. We show the algorithm converges in probability to a stationary point of the expected revenue function under mild conditions. The algorithm is relatively efficient even on large network problems, and in our simulation experiments it produces significant revenue increases relative to traditional virtual nesting methods. On a large-scale, real-world airline example using choice behavior models fit to actual booking data, the method produced an estimated 10% improvement in revenue relative to the controls used by the airline. The examples also provide interesting insights into how protection levels should be adjusted to account for choice behavior. Overall, the results indicate that choice behavior has a significant impact on both capacity control decisions and revenue performance and that our method is a viable approach for addressing the problem.

Key words: choice behavior; revenue management; network capacity control; stochastic approximation; stochastic gradients
History: Received: October 21, 2004; accepted: February 22, 2007.

1. Introduction

Optimally rationing the amount of capacity sold among various products is a central problem in airline revenue management (RM). This rationing takes place by dynamically controlling the availability of products (ticket types with different restrictions and fares) in response to factors such as the capacity and time remaining prior to departure and forecasts of future demand. The so-called *single-resource* problem involves rationing capacity on a single flight leg; network RM problems involve rationing the capacity of a network of flights among the various products (itinerary-fare-class combinations) sold on the network. The book by Talluri and van Ryzin (2004b) provides a good overview of both single-resource and network RM problems.

Recently, there has been a growing interest in more accurately modeling customer behavior in RM problems. Indeed, the traditional models of revenue management are quite simplistic in this regard; they assume demand for each product is a stochastic process that is unaffected by the availability of other

products. This so-called *independent demand model* assumption is made primarily for analytical tractability. Yet it has been long recognized that customers in fact "buy up" to higher fares if discounts are unavailable, or "buy down" to discounted fares when they are made available (see Belobaba 1987). They may also "divert" to different flight times or different routes if their preferred choice is not available. With the current industry trend toward simplified, less-differentiated fare structures (driven largely by the practices of low-cost carriers such as Jet Blue in the United States and RyanAir in Europe), the assumptions of the independent demand model are becoming even more tenuous.

Representing demand using discrete choice models of customer purchase behavior has emerged as an appealing alternative to the independent demand model, and research in this area of late has been growing. Several researchers have looked at approximate analyses of customer choice behavior for single-leg RM problems. Belobaba (1987) proposed a modification of the expected marginal seat revenue (EMSR) heuristic to account for the probability of a customer buying a higher fare when a low fare is closed (see also Belobaba and Weatherford 1996). Phillips (1994) proposed a "state-contingent" model of revenue management in which demand for products depends on the set of available products (the system "state"). Talluri and van Ryzin (2004a) provide an exact analysis of a single-leg model of RM under a general discrete choice model of demand. Their work provides a relatively clear picture of the optimal policy in the single-resource case. Recently, Boyd and Kallesen (2004) illustrate the effect of considering demand models for price-sensitive customers in a single-leg setting, where customers are price sensitive and not perfectly segmented, and therefore may end-up purchasing a fare product that costs less than they are actually willing to pay.

Modeling customer choice behavior on networks leads to considerably more difficult RM problems. Still, there is a growing body of work on choice-based network methods. To our knowledge, the earliest work to consider choice behavior in networks is the passenger origin and destination simulator (PODS) study of Belobaba and Hopperstad (1999). The main aim of this work is to understand how customer choice behavior affects traditional RM methods (primarily based on the independent demand model). An interesting industry application of choice modeling in networks is reported by Andersson (1989, 1998) and Algers and Besser (2001), who applied logit choice models to estimate buy-up and recapture factors at one of Scandinavian Airline's hubs. Zhang and Cooper (2005) analyze choice among different departure times between the same city pair (so-called "parallel flights"). Their model assumes that customers choose among the same fare class on different flights, but not among fare classes themselves (e.g., customer segments are still effectively separated by the fare class restrictions). They develop bounds and approximations to the resulting dynamic program. Gallego et al. (2004) propose and analyze a natural choice-based analog of the widely used deterministic linear programming (DLP) model of traditional network RM. This choice-based DLP model determines the amount of time to offer each possible subset of available products under assumptions of deterministic demand. Liu and van Ryzin (2008) further analyze this model and propose a dynamic programming decomposition heuristic based on it.

The work of Zhang and Cooper (2005), Gallego et al. (2004), and Liu and van Ryzin (2008) are similar in that each tries to determine (or approximate) the structure of a choice-based network capacity control policy. That is, they do not assume a policy a priori, but rather the policy structure is an output of their analysis. To achieve this, however, requires making simplifying assumptions or approximations, e.g., that demand is deterministic as in the LP model analyzed by Gallego et al. (2004) and Liu and van Ryzin (2008), or that the network consists exclusively of parallel flights with customers choosing only among alternatives within the same fare class, as in the work of Zhang and Cooper (2005).

In this paper, we take a somewhat different approach. We assume the firm uses a specific parametric policy, namely a virtual nesting control policy parameterized by a set of protection levels (one for each virtual class on each leg of the network). Virtual nesting was developed at American Airlines in the 1980s (see Smith et al. 1992), and it remains a popular network control strategy in the airline industry. (Virtual nesting is described in detail below; see also Talluri

and van Ryzin 2004b.) We then develop an efficient simulation-based method to optimize over the parameters of this policy. The approach is a direct extension of the simulation-based method originally proposed by Bertsimas and de Boer (2005) and later extended by van Ryzin and Vulcano (2006) for the network RM problem under the independent demand model.

The restriction to a particular class of policies is clearly a limitation and no claim is made on the optimality (or near-optimality) of the resulting policy as a result. But the payoff is a significant increase in generality in modeling demand. Indeed, the method applies to essentially any choice behavior and any demand process one can simulate (within the confines of our sample path description of demand). As a result, very complex choice behaviors, statistical correlations, nonstationarities, etc., can be handled. Moreover, because the optimization method only requires a "black box" (oracle) to generate sample paths of demand, it allows for a clean separation of the demand modeling and optimization modules of the overall procedure. For example, one can make essentially arbitrary changes in the model of demand and customer behavior without impacting the way the optimization algorithm functions. This level of demand modeling flexibility is likely to be a significant advantage of the method in practice. Last, although simulation-based optimization methods are notoriously computationally intensive (and ours is no exception in this regard), in our experience the algorithm runs relatively quickly—even on moderately large networks. This is due in large part to the efficiency of our sample path gradient calculations. Thus, though computationally intensive, the method appears fast enough to have good practical potential.

Applying our method to several numerical examples suggests that significant revenue gains are possible from explicitly accounting for customer choice behavior. Indeed, whereas revenue gains from improvements in optimization methods in traditional RM problems are typically on the order of 1%–3%, our examples show gains on the order of 10%–20% (or more) in revenue using our choice-based RM method relative to methods based on independent demand model assumptions. Although many of these examples are hypothetical, one was obtained from a major U.S. airline and used choice models fit to actual booking data. In this large, real-world example, we estimated that, relative to the actual controls used by the airline, the improved protection level produced by our algorithm would increase revenues 10% over the last week of the booking horizon. These results show both qualitatively and quantitatively the important impact that customer choice behavior has on RM decisions.

The remainder of the paper is organized as follows: In §2 we introduce the discrete model and its continuous approximation. In §3 we present the way we improve an initial set of protection levels through a gradient based method. Section 4 shows some numerical results, and we present our conclusions in §5.

1.1. Notation

We begin by introducing some notational conventions. For a vector $x \in \mathcal{R}^n$, x_j denotes its jth component, and x^T is the vector transpose. The unit vector is given by e_i, having a 1 in position i and 0 elsewhere. For a number a, we denote $a^+ = \max\{a, 0\}$. The letter $\mathcal{N}$ represents the set $\{1, \ldots, n\}$.

We use $\mathbf{I}\{\cdot\}$ for the indicator function, a.s. means *almost surely*; c.d.f. is short for *cumulative distribution function*; w.p.1 is short for *with probability* 1; i.i.d. is short for *independent and identically distributed*, and CI is used for *confidence interval*. Finally, the symbol $\Phi(\cdot)$ stands for the cumulative distribution function of the standard normal distribution.

2. Model Formulation

The network has m resources (flight legs), which can be used to provide n products (itinerary-fare-class combinations). Define the incidence matrix $A = [a_{ij}] \in \{0, 1\}^{m \times n}$. We let $a_{ij} = 1$ if resource i is used by product j, and $a_{ij} = 0$ otherwise. Thus, the jth column of A, A_j, is the incidence vector for product j, and the ith row, A^i, is the incidence vector for resource i. We use the notation $i \in A_j$ to indicate that resource i is used by product j, and $j \in A^i$ to mean that product j uses resource i. The revenue for accepting one unit of product $j \in \mathcal{N}$ is r_j. The state of the network is described by a vector $x^\mathsf{T} = (x_1, \ldots, x_m)$ of resource capacities. If one unit of product j is sold, the state of the network changes to $x - A_j$. To simplify the analysis, we

ignore cancelations and no-shows.[1] Another essential assumption we make is that capacity and demand are continuous quantities. We do this to produce a model that is sufficiently smooth to admit derivatives. Hence, the capacity x is continuous.

2.1. Virtual Nesting Policy

We assume the network uses a *virtual nesting* control policy, defined as follows: Each product j is mapped to virtual class $c_i(j)$ on each resource i used by product j as given by a fixed *indexing* scheme. We assume there are $\bar{c}$ protection levels in each resource (i.e., $\bar{c}+1$ virtual classes) and that the indexing $c_i(j)$ is given. In practice, a variety of heuristic methods are used for indexing (see Chapter 3 of Talluri and van Ryzin 2004b), but roughly each attempts to cluster products based on various estimates of their "net benefit" to the network (e.g., their revenue in excess of the opportunity cost of capacity they consume).

Capacity is then controlled using nested protection levels for the virtual classes on each leg (see Chapter 2 of Talluri and van Ryzin 2004b for a detailed description of nested allocation policies). Specifically, we assume that virtual classes are ordered, with virtual class 1 the highest in the nesting order, followed by virtual class 2, etc. Let y_{ic} denote the protection level for virtual classes c and higher on resource i. Again, because our model is continuous, protection levels are assumed continuous as well. Requests for virtual class $c+1$ on leg i are then accepted if and only if the remaining capacity exceeds the protection level y_{ic} for higher virtual classes $c, c-1, \ldots, 1$. In other words, virtual class $c+1$ requests only have access to the capacity in excess of y_{ic} on leg i. A request for a product j is accepted if and only if capacity is available for its corresponding virtual classes on each resource $i \in A_j$.

Let $y = (y_{11}, \ldots, y_{1\bar{c}}, \ldots, y_{m1}, \ldots, y_{m\bar{c}})$ denote the vector of all $m\bar{c}$ protection levels. Because protection levels are nested, we require that

$$0 \le y_{i1} \le y_{i2} \le \cdots \le y_{i\bar{c}} \le x_i, \quad i = 1, \ldots, m, \qquad (1)$$

where x_i is the capacity of resource i, $x_i > 0$, $\forall i$. Let Θ be the set of all y satisfying these constraints. We assume dummy protection levels y_{i0} when needed; i.e., $y_{i0} = 0$, $\forall i$, representing the fact that there is no protection level for the highest virtual class.

In the description above, we are assuming a form of nesting called *theft nesting*. Theft nesting refers to the case where protection levels y remain constant throughout the booking process. Although our approach can be modified to work with *standard nesting*—a policy in which fixed *booking limits*,[2] rather than fixed protection levels, are used—the resulting formulas are more complex. Hence, for the ease of presentation, we focus on the former. Both nesting methods are found in airline industry practice, although standard nesting is more common. See Talluri and van Ryzin (2004b) for a detailed discussion of theft versus standard nesting.

2.2. Demand Model

As mentioned, we use a very general model of demand. It is based on a sample path description of the number of customers, their arrival order and preferences. Let T denote the total number of customers in a sample path. T is assumed finite w.p.1. Each customer $t = 1, \ldots, T$ has preferences among the set of products $\mathcal{N}$, which are described by a simple ranking, $l_t = [l_{t1}, \ldots, l_{tn}]$, with $l_{tk} = j$ denoting that customer t's kth preferred choice is to purchase product j. A value $l_{tk} = 0$ denotes that customer t's kth preferred choice is to not purchase any product.

This preference list could be the result of a simple utility maximization mechanism where each customer t assigns a utility U_t^j to purchasing product j, $j \in \mathcal{N}$, and to not purchase, U_t^0. The utility itself is normally a function of various product attributes, e.g., departure day, departure time, fare paid, number of stopovers, etc. Although this description of preferences is not perfectly general (e.g., see Kreps 1988 for

[1] In practice, cancelations and no-shows are normally handled by first computing "virtual capacities"—capacities that exceed the physical capacity—on each resource so as to approximately balance the opportunity cost of excess capacity against the costs of denied service. These virtual capacities are then used as inputs to a network capacity control model, which attempts to optimally ration the virtual capacity. Hence, one can consider capacities in our model to be these virtual capacities.

[2] Booking limits are defined as the difference between the initial capacity and the corresponding protection levels. Specifically, the booking limit $b_{ic} = x_i - y_{i,c-1}$ represents the number of seats reserved for virtual class c over leg i.

a detailed treatment of utility theory and its underlying assumptions and limitations), it encompasses a wide range of choice behaviors that arguably includes most choice behaviors of practical interest.[3] To illustrate, suppose that customer t's first preference is product 3, her second preference is product 5, and beyond that she prefers not to purchase any product at all. Then we would represent her preferences as $l_t = [3, 5, 0, \ldots, 0]$.

We further assume each customer requires a continuous quantity $q_t > 0$, which is a realization of a random variable, Q_t, with support $[0, \bar{Q}]$. Customers consume products in their preference order until their desired quantity is met. Again, this demand is treated as continuous and can be viewed in fluid model terms as follows: a customer drains their most preferred fluid (product) first. If this fluid is not available or runs out, the customer drains the second most preferred fluid and so on. This process continues until either the customer's entire requirement q_t is met or all fluids valued higher than the no-purchase choice are exhausted.

To illustrate, consider again our customer t with preferences $l_t = [3, 5, 0, \ldots, 0]$. Suppose she requires $q_t = 3$ units and that there is one unit each of product 3 and 5 available for sale. (This availability is, of course, a function of the protection levels and the capacity at the time customer t arrives.) Then she would consume one unit of product 3, one unit of product 5, and not purchase any remaining products (leaving one unit of her demand unmet). Note that this is clearly not the most realistic assumption (e.g., in the airline case, our customer t might be buying for her family of three and the above sequence would imply that she would take one flight, her husband another and their child would be left behind! (or some such permutation)). However, it has the considerable advantage of making the resulting model smooth, because a small change in the available capacity of one product produces only a slight shift in customer t's consumption. For example, were the available capacity of product 3 to increase to 1.2, then customer t would consume 1.2 units of product 3, 1 unit of product 5, and let 0.8 units of her demand go unmet. In this way, changes in protection levels produce smooth changes in the quantities purchased. Although this fluid approximation could potentially distort the results of the algorithm, our simulation tests in §4 assume customers make more realistic all-or-nothing decisions. In these tests, our algorithm produces significant revenue improvements, which suggests the approximation is indeed a reasonable one.

Each sample path, therefore, is a sequence $\omega = \{(l_1, q_1), (l_2, q_2), \ldots, (l_T, q_T)\}$, which we assume to be defined on a probability space $(\Omega, \mathcal{F}, P)$. Other than the indicated assumptions above (e.g., that T is finite w.p.1 and that the random variable Q_t is continuously distributed and bounded), no assumptions about this distribution are required. Customer preferences may change over time, be correlated with each other, depend on the total demand T, etc. Indeed, in essence, all we require is an "oracle" that can generate sample paths ω drawn from some (perhaps implicitly defined) distribution. For example, the sequence and preference could be obtained via a detailed simulation of each individual customer's decision process, as in the PODS simulations of Belobaba and Hopperstad (1999).[4]

2.3. Sample Path Revenues

To describe the revenues on a sample path basis, it is convenient to introduce some extra notation. (The precise description of sample path revenues here is notationally complex, but it is conceptually quite straightforward.) For customer t with preferences l_t, define p_t as the number of nonzero entries in l_t (the number of products the customer is willing to purchase). Let $b(t, j)$ denote the rank assigned to product j by customer t. That is,

$$b(t, j) = k, \quad \text{if } j \in \mathcal{N} \text{ and } l_{tk} = j.$$

For completeness, we define $b(t, j) = p_t + 1$ if j is not in the preference list of customer t. To simplify notation, we index a product j with $[k]$ when $b(t, j) = k$

[3] The approach is essentially the same as that used by Mahajan and van Ryzin (2001, §2.2.1). They show how the multinomial logit model, Markovian second choice, universal backup, Lancaster demand, and independent demand are some special cases of a utility maximization mechanism.

[4] In the PODS model, customers are assumed to have a given travel objective with time window constraints on their departure and arrival times, valuations for their preferred airline, time/price sensitivities, etc.

(i.e., when j is the kth preferred choice). Continuing our example, if $l_t = [3, 5, 0, \ldots, 0]$, then $b(t, 3) = 1$, $b(t, 5) = 2$, and $p_t = 2$.

The amount of capacity of product j purchased by customer t is the minimum of the customer's residual unmet demand (their original demand q_t minus the quantity purchased of more preferred products) and the capacity available for product j. Let $x_i^j(t)$ denote the capacity available to customer t in product j's virtual class on resource i, which is the remaining capacity $x_i(t)$ minus the protection level for virtual classes higher than the virtual class of j on resource i, less the amount of capacity already purchased of products with higher preference that also use resource i. The column vector representing the acceptance function for customer t is denoted $u(x(t), y, l_t, q_t) \in \mathcal{R}_+^n$, where the jth component corresponds to consumed quantity of product j. In symbols,

$$x_i^j(t) = \Bigg(x_i(t) - y_{i,\, c_i(j)-1} - \sum_{k=1}^{b(t,\,j)-1} [u_{[k]}(x(t), y, l_t, q_t)\mathbf{I}\{i \in A_{[k]}\}]\Bigg)^+, \quad (2)$$

where the sum is defined to be empty when $b(t, j) = 1$.

The formal definition for the capacity allocated to a product j is then

$$u_j(x(t), y, l_t, q_t) = \begin{cases} \min\left\{q_t - \sum_{k=1}^{b(t,\,j)-1} u_{[k]}(x(t), y, l_t, q_t),\right. \\ \qquad \left. x_i^j(t)\colon i \in A_j\right\}, & \text{if } 1 \le b(t, j) \le p_t; \\ 0, & \text{if } b(t, j) = p_t + 1. \end{cases} \quad (3)$$

In words, if product j is the customer's first preference, then we drain as much of it as possible (up to the customer's total demand q_t). If j is not the first preference, we first drain from the more preferred options first and then satisfy any residual demand as much of this as possible from j. When $b(t, j) = p_t + 1$, this implies that the no-purchase option is preferred to product j, so product j is not consumed at all. (Note that this means that the demand q_t may be not fully satisfied as discussed above.)

Define $R_t(x(t), y, \omega)$ to be the revenue-to-go over periods $t, t+1, \ldots, T$, starting with a vector $x(t)$ of remaining capacities and protection levels y. We then have the following set of recursive *forward equations* for determining the revenues

$$R_t(x(t), y, \omega) = r^{\mathsf{T}} \cdot u(x(t), y, l_t, q_t) + R_{t+1}(x(t+1), y, \omega) \quad (4)$$

$$x(t+1) = x(t) - A \cdot u(x(t), y, l_t, q_t); \quad (5)$$

for $t = 1, \ldots, T$, with boundary conditions

$$x(\tau + 1) = 0$$

$$R_{T+1}(x, y, \omega) = 0 \quad \text{for all } x, y, \omega; \quad \text{and}$$

$$x(1) = x.$$

The total sample path revenue is given by

$$R(y, \omega) = R_1(x, y, \omega). \quad (6)$$

Our objective is to maximize the expected revenue function, $g(y) = \mathrm{E}[R(y, \omega)]$, over the set Θ of feasible protection levels:

$$\max_{y \in \Theta} g(y). \quad (7)$$

Here, and in what follows, expectation is taken with respect to the random sample path ω.

3. Stochastic Approximation Algorithm

Our computational approach is a generalization of that used by van Ryzin and Vulcano (2006). Indeed, if $p_t = 1$ for all customers in our model (i.e., all customers have only one preferred product), then we in fact recover the problem studied by van Ryzin and Vulcano (2006). The technical details of the approach are given in the appendix and in the online appendix;[5] here, we only give a high-level overview of the essential ideas.

3.1. Algorithm Overview

The first key idea of the algorithm is to differentiate (4)–(5) to obtain an efficient recursion for computing the sample path gradients $\nabla_x R_1(x, y, \omega)$ and $\nabla_y R_1(x, y, \omega)$. (Some additional "smoothing" of the recursion using a random perturbation of capacity is required to ensure these sample path gradients exist w.p.1.; see the appendix for technical details.)

[5] An online appendix to this paper is available on the *Manufacturing & Service Operations Management* website (http://msom.pubs.informs.org/ecompanion.html).

Although these gradient calculations are complex to express algebraically, they are simple and efficient to implement algorithmically. Online Appendix C provides a complete pseudocode for this calculation along with a complexity analysis, which shows that computing the entire gradient has essentially the same complexity as simulating the sample path in the first place. The differentiability properties and efficiency of this recursion are the main payoffs for the lack of realism introduced by the continuous capacity and demand assumptions.

As shown in online Appendix B, because the revenue function $R_1(x, y, \omega)$ is Lipschitz continuous in x, y, we can justify the interchange of differentiation and expectation; hence,

$$\nabla_y g(y) = \nabla_y \mathrm{E}[R(y, \omega)] = \mathrm{E}[\nabla_y R(y, \omega)].$$

Therefore, $\nabla_y R(y, \omega)$ is an unbiased estimator of $\nabla_y g(y)$. This stochastic gradient can then be used in place of the actual gradient in a steepest ascent type algorithm to search for a stationary point y of $g(\cdot)$. This is the essential idea behind stochastic approximation (SA), a method originating in the work of Robbins and Monro (1951). Kushner and Clark (1978), Benveniste et al. (1990), and the recent book by Kushner and Yin (2003) contain expositions of its theory.

To maximize $g(y) = \mathrm{E}[R(y, \omega)]$ over the convex compact set Θ defined by constraint set (1), we require an initial feasible point $y^{(0)} \in \Theta$, and a sequence of step sizes $\{\rho^{(k)}\}$ satisfying

$$\rho^{(k)} > 0, \quad \lim_{k\to\infty} \rho^{(k)} = 0, \quad \sum_{k=1}^{\infty} \rho^{(k)} = +\infty, \quad \text{and}$$
$$\sum_{k=1}^{\infty} |\rho^{(k)}|^2 < +\infty. \tag{8}$$

We used step sizes $\rho^{(k)} = a/k$, where $a > 0$ is a constant (chosen in our case based on experimentation with the method). For simulated demand streams $\omega^{(1)}, \ldots, \omega^{(N)}$, the stochastic gradient method proceeds as follows:

Stochastic Gradient Algorithm.

Step 1. Compute an initial feasible set of protection levels $y^{(0)}$.

Step 2. For $k := 1$ to N do:

(a) Calculate the sample path gradient over demand stream $\omega^{(k)}$: $\nabla_y R(y^{(k-1)}, \omega^{(k)})$.

(b) Set new step size $\rho^{(k)} := a/k$.

(c) Update the protection levels for the next iteration, using the equation

$$y^{(k)} := \Pi_\Theta(y^{(k-1)} + \rho^{(k)} \nabla_y R(y^{(k-1)}, \omega^{(k)})),$$

where $\Pi_\Theta(\cdot)$ is the orthogonal projection into the feasible set Θ.

Step 3. Return $y^{(N)}$. Stop.

Some comments about our implementation choices are in order. First, we ran a fixed number, N, of iterations to improve an initial feasible set of protection levels obtained in Step 1 (typically, N was on the order of thousands). Alternatively, various stopping criterion could be employed to terminate the algorithm, although one weakness of stochastic gradient methods is that they lack good stopping criteria (Shapiro 2000). Second, the step size chosen in Step 2(b) is a simple and popular choice. Alternative step size rules for more general stochastic quasigradient methods can be found in Pflug (1988). Third, note that the projection in Step 2(c) is of the form:

$$y = \Pi_\Theta(z) \quad \Leftrightarrow \quad y = \arg\min_{y' \in \Theta} \|y' - z\|.$$

For each resource i, this projection is given by a quadratic program with linear constraints, which can be solved efficiently using standard methods like barrier-type algorithms (see Bertsekas 1999, Chapter 4). This projection typically involves a small number of variables and a small number of constraints. Last, we note that in a commercial implementation the simulations could be run on parallel processors, with each CPU generating its own sequence of demand and calculating the resulting sample path gradient. In this sense, the algorithm is highly parallelizable.

3.2. Convergence

Theorem B1 in online Appendix B shows that the revenue function of our model is not quasiconcave in general; hence, our SA algorithm is unlikely to be globally convergent. However, it has at least robust local convergence properties. This convergence is based on a smoothed version of the problem presented in Appendix A. Here we only state the main result. Recall that the gradient $\nabla_y R(y^{(k-1)}, w^{(k)})$ is a noisy representation of the gradient of $g(y^{(k-1)})$. Let the noise

(error) in the gradient at iteration k be the vector $\xi^{(k)} \equiv \nabla_y R(y^{(k-1)}, w^{(k)}) - \nabla g(y^{(k-1)})$. From Lemma B1 in online Appendix B, $E[\xi^{(k)} \mid y^{(0)}, \ldots, y^{(k-1)}] = 0$, w.p.1. Let the cumulative step sizes be defined as $s_k = \sum_{i=1}^{k-1} \rho^{(i)}$, and define a function $m(s)$ such that $m(s) = \max\{k\colon s_k \le s\}$ for $s \ge 0$, and $m(s) = 0$ otherwise. Suppose the following conditions hold:

Condition 1 (C1). $\{\rho^{(k)}\}$ is a sequence of positive real numbers such that $\rho^{(k)} \to 0$, $\sum_{k=1}^{\infty} \rho^{(k)} = +\infty$.

Condition 2 (C2). Let the constraint set for the problem be defined by $\Theta = \{y\colon \theta_j(y) \le 0,\ j = 1, \ldots, z\}$. The set Θ is closed and bounded. The $\theta_j(\cdot)$, $j = 1, \ldots, z$, are continuously differentiable. At each y that is on the boundary of Θ, the gradients of the active constraints are linearly independent.

Condition 3 (C3).

$$\lim_{k\to\infty} P\left(\sup_{m(s_k+s)\ge l\ge k} \left| \sum_{i=k}^{l} \rho^{(k)} \xi^{(k)} \right| > \epsilon \right) = 0$$

for each $\epsilon > 0$ and $s > 0$.

Condition 4 (C4). $g(\cdot)$ is a continuously differentiable real valued function.

Condition 5 (C5). $\rho^{(k)} E|\xi^{(k)}|^2 \to 0$ as $k \to \infty$.

Then, we have:

Theorem 1. *Let Θ^* be the set of Kuhn-Tucker points for the continuous problem* (A1)–(A2) *stated in Appendix A. Then, the stochastic gradient algorithm described above verify Conditions* 1–5. *Moreover, if Θ^* is a connected set, the sequence of points $y^{(k)} \to \Theta^*$ in probability as $k \to \infty$.*

Proof. The proof follows from properties of the revenue function discussed in online Appendix B. C1 is satisfied by our choice of the step sizes $\rho^{(k)}$ in (8). C2 is satisfied by our constraint set (1). C3 holds for our gradient estimator by choice of $\rho^{(k)}$, boundedness of $\nabla_y R(y^{(k-1)}, \omega)$, $\xi^{(k)}$, and $\nabla g(y)$, and by Lemma B2 in online Appendix B. C4 holds by Theorem B2 in online Appendix B. C5 holds by choice of $\rho^{(k)}$ and Lemma B2 in online Appendix B. The convergence result follows from Theorem 6.3.1 in Kushner and Clark (1978). □

A weaker convergence result holds when Θ^* is not connected. To show it, we first define an interpolation for the sequence $\{y^{(k)}\}$. Define the continuous function $y(s)$ by:

$$y(s) = \begin{cases} y^{(k)} & \text{if } s = s_k \\ \dfrac{s_{k+1} - s}{\rho^{(k)}} y^{(k)} + \dfrac{s - s_k}{\rho^{(k)}} y^{(k+1)} & \text{if } s \in (s_k, s_{k+1}). \end{cases}$$

Observe that $y(s)$ is just a linear interpolation of the values $y^{(k)}$ as a function of the cumulative step sizes s_k. Let $N_\epsilon(\Theta^*)$ denote the epsilon neighborhood of the set Θ^*. Kushner and Clark (1978, Theorem 6.3.1) show that under Conditions 1–5, if Θ^* is not connected, then for each $\delta > 0$ and $\epsilon > 0$, there exists a finite constant s_0 such that $s > s_0$ implies

$$\lim_{k\to\infty} P\left(\frac{1}{2s} \int_{-s}^{s} \mathbf{I}\{y(s_k + v) \in \mathcal{R}^{m\times\bar{c}} \backslash N_\epsilon(\Theta^*)\}\, \mathrm{d}v \ge \delta \right) \le \delta.$$

Roughly, this result says that the "the average amount of time" the iterates $y^{(k)}$ lie more than ϵ away from a point in Θ^* (averaging over a sufficiently large but finite interval) becomes arbitrarily small as k increases. It is basically a convergence in probability of a "moving average" of $y^{(k)}$ rather than a convergence of $y^{(k)}$ itself.

Summarizing, the algorithm we proposed satisfies the conditions for the convergence to a Kuhn-Tucker point. When Θ^* is connected, we have convergence in probability. Even if Θ^* is not connected, we still have a guarantee of convergence of the average of iterates to a point arbitrarily close to Θ^*. We emphasize again, though, that all these are only local convergence guarantees.

4. Numerical Experiments

In this section, we illustrate our method on several numerical examples. These examples both give a sense of the revenue improvements obtainable by accounting for choice behavior as well as a qualitative understanding of the differences in the capacity control decisions that result. We start with several small examples, where it is easy to see intuitively how (and why) the algorithm modifies the initial protection levels. The later examples are larger networks in which it is quite difficult to intuitively understand the changes made by the algorithm. The last example is based on actual airline data and gives a sense of the real-world performance of the method. Collectively, these examples illustrate the potential revenue improvements and also give a sense of the computation time required by the algorithm.

We implemented the stochastic gradient algorithm in C++, and ran our experiments on a Pentium IV Workstation (CPU of 2.00 GHZ, and RAM of 512 Mb),

under Windows 2000.[6] For the computation, we used a step size of $\rho_k = 0.9/k$ in Step 2(c). Kleywegt and Shapiro (2001) point out that methods like ours are very sensitive to the choice of the step sizes; small step sizes result in very slow progress towards the optimum, while large step sizes make the iterations quite volatile. We have tried with $\rho_k = a/k$, with $a = 0.1, 0.5, 0.8, 0.9, 1.0,$ and 1.5. The best results were obtained in general with $a = 0.9$.

As for the gradient estimate, in Appendix A we perturb the remaining capacity in our original problem by a random noise term to smooth out the revenue function for theoretical convenience. However, in our computational test we did not implement this perturbation, which, as a practical matter, does not significantly affect the recursion. (See van Ryzin and Vulcano (2006, §4.1) for further discussion of this issue.) Pseudocode for our gradient recursive calculation for the special case of parallel flights is provided in online Appendix C.

For Examples 1 through 5, we generated the preference lists of arriving customers as follows: First, we consider each arriving customer to belong to one of several different *customer types* (or segments), where each type is characterized by a given preference order for the products. Aggregate demand from each customer type j is assumed normal with mean μ_j and standard deviation $\sigma_j = \sqrt{\mu_j}$. This distribution was then truncated between 0 and $2\mu_j$ to avoid negative demand. From the demand parameters, we then calculated discrete distributions (a probability mass function is calculated from the normal c.d.f.). One thousand streams of demand were simulated offline (using MATLAB, from MathWorks, Inc.), and the stochastic gradient algorithm was applied starting from an initial set of protection levels, computed as described in the examples below.

Once the protection levels were computed, we compared the decisions made under the original and improved protection levels in coupled simulations runs (common random numbers). Although our model assumes customers have continuous demand and are willing to partially purchase products, in the simulations we assumed unit demand requests and that customers buy only integral numbers of seats (all-or-nothing purchase behavior). Protection levels were not reoptimized during the booking process.

Example 1. This first example is a simple illustration of *buy-up* behavior. It considers the simplest case of a single-leg flight with two fare classes. Revenues are $r = (\$200, \$100)$, with capacity $x = 100$. There are three types of customers: Type 1 are only willing to buy the low fare, Type 2 are only willing to buy the high fare, and Type 3 are "buy-up" customers—customers whose first preference is the low fare class, but are willing to pay the high fare if it is the only choice offered. Demands for the customer types have means $\mu = (50, 10, 50)$. Types 1 and 3 arrive first, in random order, followed by Type 2 customers. Note that $\mu_1 + \mu_3 = x$.

Applying Littlewood's (1972) rule[7] by aggregating both low fare Types 1 and 3 we get an initial protection level of $y^{(0)} = 10$. Note that again the ratio r_1/r_2 equals 0.5, giving a protection level for the high class equal to its demand mean. The stochastic algorithm then brings this protection level up to $y^{(N)} = 100$, i.e., up to the entire capacity of the flight. In other words, only the high fare is offered for this flight. The intuition here is that if we offer the low fare then Type 3 customers will end up paying just \$100 when they are willing to pay \$200. This new protection level is in fact optimal according to the buy-up formula presented by Belobaba and Weatherford (1996), which is exact in the simple two-class case.[8]

The revenue obtained rises from \$10,993 up to \$12,003, representing an increase of 9.18% with a 95%

[6] We used Microsoft Visual C++ 6.0 to build a Win32 console application. We linked our code with a LINDO application programming interface (Lindo Systems, Inc.) to make the projection in Step 2(c). This routine uses a barrier-type algorithm to solve the resulting quadratic program.

[7] Recall that Littlewood's rule (see Littlewood 1972) establishes that if the cumulative distribution of the demand D_1 for the high fare class is continuous, then the optimal protection level y_1^* is given by the solution to the simple expression: $r_2 = r_1 P(D_1 > y_1)$.

[8] The buy-up formula says that for high class demand D_1 and protection level y_1, it is optimal to accept class 2 as long as

$$r_2 \geq (1-s) r_1 P(D_1 > y_1) + s r_1,$$

where s is the probability that a customer for class 2 will buy a class 1 fare if class 2 is closed. The optimal protection level is the y_1 such that the formula is verified for equality. In our case, $s = \mu_3/(\mu_3 + \mu_1) = 0.5$, $r_1 = 200$, and $r_2 = 100$, leading to $y_1^* = \infty$, or practically, $y_1^* = x$.

CI of $(-2.39\%, 20.76\%)$. In this case, the load factor drops from 1.0 down to 0.60. This occurs because we lose all Type 1 customers (those only willing to pay the low fare), yet increase revenues by forcing Type 3 customers to pay the high fare.

Example 2. This next example is a simple network with two parallel single-leg flights between a given origin–destination (O–D) pair—a 7 A.M. flight and a 1 P.M. flight. (This is the sort of network studied by Zhang and Cooper 2005.) We consider two classes (high and low) per flight (i.e., there are $n = 4$ products), with $r = (\$200, \$100)$, and aircraft capacity of $x_1 = x_2 = 100$.

There are three types of customers: Type 1 customers are customers who can qualify for the low fare (e.g., leisure customers). They have a preference for the early morning flight, but are willing to take the afternoon flight (i.e., their first choice is the low fare class on the first flight, and the second choice is the low fare class on the afternoon flight). Type 2 and 3 customers cannot meet the restrictions of the low fares and have a strong time preference (e.g., business travelers). Type 2 only wants the early morning flight and Type 3 only wants the afternoon flight. Mean demands for the customer types are $\mu_1 = 100$, $\mu_2 = 30$, and $\mu_3 = 10$. The Type 1 customers arrive first, followed by Types 2 and 3 (in randomly mixed order of arrival).

The initial set of protection levels was calculated using Littlewood's (1972) rule disregarding the choice behavior. Given that the ratio $r_2/r_1 = 0.5$, the protection level for the high class in each flight was set at the mean for the high class demand: $y_{11}^{(0)} = 30$, $y_{21}^{(0)} = 10$. This would be the optimal policy if there was no choice behavior.

We then applied the stochastic gradient algorithm using these protection levels as a starting point. The algorithm produced new protection levels of $y_{11}^{(N)} = 47$, $y_{21}^{(N)} = 18$. Note, these are significantly higher than the protection levels recommended by Littlewood's (1972) rule. Intuitively, this is because there is a high likelihood that rejected low fare demand on one flight will be recaptured on the other parallel flight. The expected revenue increased by 2.32% (from 17,562 to 17,970) with a 95% CI for the gap of $(-4.52\%, 9.17\%)$. The load factor slightly increased from 0.69 up to 0.70.

The increase in protection levels produced by our algorithm is quite intuitive here. Littlewood's (1972) rule assumes that if we reject a low fare demand, we lose \$100. This loss is worth taking if the expected revenue from reserving the marginal seat for a high-fare customer exceeds \$100. But if low-fare customers are willing to take an alternate flight (as they are in this example), then rejecting a low fare demand on one flight does not necessarily result in a loss of revenue; rather, the customer may simply chose the other departure time (i.e., they may be "recaptured" on another flight). Hence, the expected loss is, in reality, less than \$100. (Exactly how much less is hard to say, but this is what our algorithm implicitly computes.)

Example 3. This next example is also a two-parallel-flight network with 7 A.M. and a 1 P.M. flights. However, now there are three classes per flight: high fare (HF), middle fare (MF), and low fare (LF), with $r = (\$100, \$70, \$50)$, giving a total of six products. To simplify notation, products are labeled HF, MF, or LF, followed by the departure time (e.g., LF7AM is the low fare class for the 7 A.M. flight). (In the tables below, we have omitted listing the no-purchase option in the preference vector for simplicity.) We take $x_1 = x_2 = 100$. The booking horizon is divided into four periods, where period 1 is the earliest one, and period 4 is the closest to the departure date. The customer behavior is also different. All customers are able to purchase the low fares and differ only in their willingness to pay higher fares and their preference for departure time. Details of each customer type and the demand in each period are given in Table 1. This example is motivated by the undifferentiated fare structures offered by low-cost carriers flying point-to-point routes (e.g., JetBlue). For these carriers, customers always choose the lowest available price and the problem is effectively one of dynamic pricing.

The initial set of protection levels for the morning flight was chosen as $y_1^{(0)} = (25, 65)$ based on a simple mean-demand calculation: customer Type 5 (fifth row in Table 1) is eventually willing to pay the full fare and, hence, 25 seats are reserved for this type (its mean demand is 25); customer Type 3 is eventually willing to pay the middle fare and, hence, an additional 40 seats are reserved for this type (its mean demand is 40). Analogously, $y_2^{(0)} = (30, 65)$.

Table 1 Description of Customer Types for Example 3

Description	Type (preference order)	Mean demand	Booking periods
Price sensitive, early preference	[LF7AM, LF1PM]	50	1, 4
Price sensitive, late preference	[LF1PM, LF7AM]	40	1, 4
Buy-up, early preference	[LF7AM, LF1PM, MF7AM, MF1PM]	40	2
Buy-up, late preference	[LF1PM, LF7AM, MF1PM, MF7AM]	35	2
Price insensitive, morning	[LF7AM, MF7AM, HF7AM]	25	2, 3
Price insensitive, afternoon	[LF1PM, MF1PM, HF1PM]	30	2, 3

After applying the stochastic gradient algorithm, these protection levels were changed to $y_1^{(N)} = (18, 90)$ and $y_2^{(N)} = (40, 62)$. Note that the controls reserve more seats for the high class on the afternoon flight, and more seats for the middle class on the morning flight. This is taking advantage of the fact that more customers with late departure time preferences are willing to buy-up, whereas relatively fewer customers with early departure time preferences are willing to pay more for their preferred time. The load factor is maintained at 0.86 after applying the stochastic algorithm. The total revenue from the change in policy is quite clear indeed: revenues increase by 5.34% (from \$11,374 to \$11,982 with a gap 95% CI of (0.77%, 9.92%)).

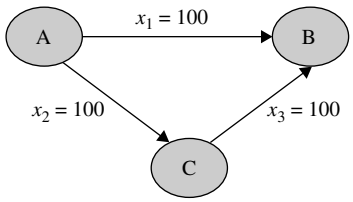
Example 4. This is a slightly larger parallel flight network. It consists of four flights serving a given O–D pair with different departure times (7 a.m., 10 a.m., 1 p.m., and 4 p.m.). The capacity for the flights is $x_i = 100$, $i = 1, \ldots, 4$. Each flight has two fare classes with revenues $r = (\$200, \$100)$, creating $n = 8$ products in total. We consider 12 customer types, arriving during three stages of the booking process as detailed in Table 2. Within each stage, customer types are randomly mixed.

Initial protection levels were computed by Littlewood's (1972) rule disregarding the choice behavior (i.e., just considering the first preference of each customer type and then aggregating these types by product to get the mean demands). This resulted in the protection levels $y^{(0)\mathsf{T}} = (20, 17, 15, 25)$. In words, there are 20 seats reserved for class 1 in the 7 a.m. flight, 17 in the 10 a.m. flight, 15 in the 1 p.m. flight, and 25 in the 4 p.m. flight. This produced an expected revenue of \$47,270 with a load factor of 0.99. Our stochastic gradient algorithm produced the set of protection levels $y^{(N)\mathsf{T}} = (60, 47, 39, 65)$ and generated an expected revenue of \$58,923 and a load factor of 0.97. This represents a revenue gain of 24.65% with a 95% CI of (18.24%, 31.06%). To see intuitively why the algorithm produces these protection levels, note that if we aggregate the mean demand from those customers willing to pay the high fare (the buy-up and full fare customer types), we almost get the same protection levels: (60, 47, 40, 65). This is because the ratio between the low and high fares is 0.5; hence, it is

Table 2 Description of Customer Types for Example 4

Order of arrival	Description	Type (preference order)	Mean demand
Price sensitive (1st)	Early preference	[LF7AM, LF10AM, LF1PM, LF4PM]	40
	Middle (early) preference	[LF10AM, LF1PM, LF7AM, LF4PM]	80
	Middle (late) preference	[LF1PM, LF10AM, LF4PM, LF7AM]	80
	Late preference	[LF4PM, LF1PM, LF10AM, LF7AM]	60
Buy-up (2nd)	Early preference	[LF7AM, HF7AM]	40
	Middle (early) preference	[LF10AM, HF10AM]	30
	Middle (late) preference	[LF1PM, HF1PM]	25
	Late preference	[LF4PM, HF4PM]	40
Full fare (3rd)	Early preference	[HF7AM]	20
	Middle (early) preference	[HF10AM]	17
	Middle (late) preference	[HF1PM]	15
	Late preference	[HF4PM]	25

Figure 1 Network for Numerical Example 5, with Three Legs and Two Classes Per Leg



Note. Protection levels are for scenario 1.

reasonable to set protection levels equal to the mean demand of all those customers willing to pay the high fare. Note, however, that the mean demand heuristic that works well here can also perform significantly worse than the algorithm (see Example 3).

Example 5. This example is based on the small network of Figure 1 and illustrates a simple case of path choice between a connecting and nonstop flight. The are $m = 3$ resources, with capacity $x_i = 100$, and 2 classes per resource. We consider $n = 7$ products labeled with a prefix HF or LF (high or low fare, respectively) followed by the cities involved in the itinerary (e.g., LF-ACB represents the low fare class for the itinerary joining cities A and B with a connection at C). The products are listed in Table 3.

There are 10 customer types arriving over three booking periods. The booking periods are labeled from the earliest to the latest. We also consider two different demand scenarios. The behavioral description of each customer type, their booking order, and mean demands under each scenario are specified in Table 4.

For both scenarios, we use the *displacement-adjusted virtual nesting* (DAVN) scheme as described by Williamson (1992) to define the indexing and compute an initial set of protection levels. The method solves a deterministic linear program based on the mean demand of the first choice of all customer types. The dual variables from this linear program are used

Table 3 Description of Products for Example 5

Product	Revenue ($)	Product	Revenue ($)
HF-AB	300	HF-CB	200
LF-AB	180	LF-CB	100
HF-AC	200	LF-ACB	130
LF-AC	100		

Table 4 Description of Customer Types for Example 5

Type (preference order)	Description	Booking order	Demand mean Scenario 1	Demand mean Scenario 2
[HF-AB]	Only want direct flight	3	30	10
[LF-AB, LF-ACB]	Moderately prefer direct	1	60	10
[LF-ACB, LF-AB]	Price sensitive	1	20	30
[HF-AC]	High fare, AC	3	10	30
[HF-CB]	High fare, CB	3	10	10
[LF-AC, HF-AC]	AC buy-up	2	10	30
[LF-CB, HF-CB]	CB buy-up	2	10	10
[LF-AC]	AC price sensitive	1	10	60
[LF-CB]	CB price sensitive	1	10	10
[LF-AB, HF-AB]	AB buy-up	2	30	20

to compute displacement-adjusted revenues and cluster products into virtual classes. Then, a single-leg stochastic model is solved to determine protection levels for the virtual classes (in our case, Belobaba's EMSR-b heuristic (Belobaba and Weatherford 1996)). A detailed description of the implemented version of DAVN can be found in our previous paper. (See van Ryzin and Vulcano 2006, Appendix C.)

Scenario 1. The first scenario corresponds to a situation in which flight A-B is congested and we would like to force traffic onto the connecting route A-C-B. Note that direct flight A-B is congested because of demand from the first, second, and last customer types (the sum of their mean demands based on their first preference gives 120, whereas capacity is $x_1 = 100$). However, the second customer type (which has the highest mean demand) is willing to switch to the connecting flight A-C-B.

The initial solution provided by the DAVN algorithm was $y^{(0)\mathsf{T}} = (29, 9, 9)$ for the three legs, respectively, with two virtual classes per leg. The expected revenue produced is \$32,058. After applying our stochastic gradient algorithm, we obtain revised protection levels of $y^{(N)\mathsf{T}} = (68, 42, 43)$, which produce an expected revenue of \$38,381. The improvement in expected revenue is 19.72% with 95% CI (5.83%, 33.60%), and the network load factor rises from 0.67 to 0.79. Intuitively, the increase in the protection levels forces the second type of customer onto the connecting flight (thereby increasing the load factor) and also induces the buy-up types to pay the full fare. Both effects increase revenues.

Scenario 2. The second scenario illustrates a case where there is little demand for the direct flight A-B,

but the connecting flight A-C is congested due to high local demand. The initial solution provided by the DAVN algorithm was

$$y^{(0)} = \begin{pmatrix} 9 & - \\ 29 & - \\ 10 & 34 \end{pmatrix}$$

for the three legs, respectively (i.e, there are two virtual classes in legs 1 and 2, and three virtual classes in leg 3), leading to an expected revenue of \$27,112. After applying the stochastic algorithm, we obtained new protection levels of

$$y^{(N)} = \begin{pmatrix} 9 & - \\ 65 & - \\ 10 & 34 \end{pmatrix},$$

producing an expected revenue of \$31,237. The improvement in this case is 15.22% with a 95% CI of (5.66%, 24.78%). The network load factor decreased, however, from 0.67 to 0.64. This is because the customer mix changed, especially in the A-C leg, where more room is reserved for the high fare class. Hence, here most of the revenue gain is obtained by forcing buy-up of local customers on the A-C leg, which increases revenues but lowers load factors.

Example 6. This example is based on a real-world data set for a collection of flights between New York (with departures from LGA and JFK) and a major airport in Florida. It was obtained from a U.S. commercial airline as part of a sponsored research study into choice-based revenue management. For confidentiality reasons, we can only provide summary information about the example, but the network and choice behavior estimates are all based on actual commercial data. Hence, this example provides a good test of the feasibility and performance of our method on a large-scale, real-world problem.

The market considered had 320 products (itinerary fare classes) and 20 flight legs. The airline used 16 (virtual) classes on each flight. In our tests here, we focused only on the booking decisions during the last week prior to departure. This reduced the computation and estimation complexity and, moreover, was considered the most critical time period for making revenue-management decisions. Given that there are few seats left (between 1 and 41) during the last week, there are also few classes open (from 1 to 5 per flight).

Customer choice behavior was estimated using a multinomial logit (MNL) model that included attributes such as flight arrival time, fare paid, departure day, and origin airport (customers can choose between LGA and JFK). This MNL model was fit to booking data using maximum-likelihood methods. Details of the estimation method and estimation results are presented in Vulcano et al. (2008). The booking data were collected during the peak spring break travel period of 2005. Both arrival rates and the MNL model were fit to these data. We then used the fitted model to simulate sequences of customer arrivals and their preferences for 1,000 instances of the network booking process. This resulted in 480,497 different preference list orders. The instances consisted of an average of 572 arrivals during the last week prior to departure. We assumed that each customer required a single seat because our data did not have information on group sizes.

The initial set of protection levels (described in Table 5) were the protetion levels actually used by the airline as computed by a their current revenue-management system. Table 6 describes the improved protection levels computed by our algorithm.

Comparing Tables 5 and 6, we see that the most significant difference is that the lowest classes are kept closed (i.e. protection levels $5, 6, \ldots, 15$ remain unchanged). For the highest classes, there are changes in some of the legs. For example, over leg 1, class 3 is closed (namely, y_{12} equals capacity), and more seats are protected for the highest class. Over leg 2, classes 3 and 4 become slightly opened. There are minor changes over legs 6 and 8. Leg 11 seems to have been underprotected because just the highest class is open with the improved protection levels. Leg 12 is more closed now, whereas legs 13 to 16 become more open. Finally, leg 20 becomes available just for its highest class. Note that there is no clear pattern in terms of the change in the protection levels, and all the substitution effects that underlie the choice model are captured by our stochastic gradient algorithm when computing the new set of protection levels.

Comparing the improved protection levels to the original ones for this market, our simulations showed

Table 5 Initial Protection Levels for Example 6

	Protection levels Y_i															
Leg	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Remaining capacity
1	2	7	13	13	13	13	13	13	13	13	13	13	13	13	13	13
2	0	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
8	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
9	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
10	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
11	0	0	2	21	21	21	21	21	21	21	21	21	21	21	21	21
12	14	27	32	40	41	41	41	41	41	41	41	41	41	41	41	41
13	23	33	33	33	33	33	33	33	33	33	33	33	33	33	33	33
14	8	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
15	0	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
16	5	22	25	25	25	25	25	25	25	25	25	25	25	25	25	25
17	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
18	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
19	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
20	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

a revenue improvement of 10.3%, from \$46,917 to \$51,750, with a 95% CI for the gap of (4.91%, 15.69%). This is a very large increase in practical terms. The load factor was slightly increased from 0.88 to 0.90.

For this example, our algorithm took 95.7 seconds to run. Although we are not considering the time for generating each sample paths, as mentioned above in a commercial implementation, this simulation could

Table 6 Improved Protection Levels for Example 6

	Protection levels Y_i															
Leg	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Remaining capacity
1	11	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
2	0	1	5	7	7	7	7	7	7	7	7	7	7	7	7	7
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
8	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
10	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
11	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21
12	27	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41
13	27	32	33	33	33	33	33	33	33	33	33	33	33	33	33	33
14	2	10	12	13	13	13	13	13	13	13	13	13	13	13	13	13
15	2	11	11	12	12	12	12	12	12	12	12	12	12	12	12	12
16	4	16	20	25	25	25	25	25	25	25	25	25	25	25	25	25
17	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
18	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
19	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
20	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

be run on parallel processors with each CPU generating its own sequence of demand and calculating the resulting sample path gradient.

Overall, the example shows that our method works efficiently and effectively on real-world-size problem. Again, full details on this study are contained in a forthcoming paper.

5. Conclusions

We have proposed a model and method to find locally optimal nested protection levels for network capacity control under a very general model of customer choice behavior. The overall approach is appealing for two main reasons. First, the stochastic process that characterize the demand and choice processes can be completely general. Indeed, almost any simulation model of customer behavior can be used to generate the requisite sequences of customer and their preferences, which drive the optimization algorithm. This provides a great degree of flexibility in modeling customer behavior and allows for a clean separation of the choice modeling and optimization parts of the method, which is a very desirable feature in practice. Second, the proposed algorithm is easy to implement, relatively fast even when applied to large networks, and has shown promising improvement in revenue in all our computational tests. In this sense, the method appears to have the computational properties necessary for practical implementation.

Acknowledgments

The authors would like to thank Richard Ratliff and Wassim Chaar, from Sabre Holdings, for helpful feedback on earlier drafts of this work. They also thank two anonymous referees and a senior editor for their constructive comments and suggestions.

Appendix A. Smoothing the Revenue Function

We analyze the sample path revenue function $R_t(x(t), y, \omega)$ as a function of the protection levels y. In particular, by allowing partial acceptance of requests, the function $u_j(x(t), y, l_t, q_t)$ defined by (3) is continuous and piecewise-linear in y. Assuming $p_t = 1$ for all requests t, then $y_{i,\,c_i(j)-1} = x_i(t) - q_t$ and $y_{i,\,c_i(j)-1} = x_i(t)$ are points of nondifferentiability, which makes $R(y, \omega)$ a continuous but nonsmooth function of y. Indeed, we cannot even guarantee that $R_t(x(t), y, \omega)$ is differentiable with respect to y w.p.1, because the event $y_{i,\,c_i(j)-1} = x_i(t)$ can occur with some positive probability (e.g., with positive probability we can get a sequence of high-quantity requests such that the value $u_j(x(t), y, l_t, q_t) = 0$ in (3) for a sequence of consecutive ts is determined by the fact that $y_{i,\,c_i(j)-1} = x_i(t)$). This fact violates well known sufficient conditions for the differentiability of $g(y)$, and in particular for interchanging differentiation and expectation. (See Glasserman 1994 for a good reference on this topic.)

To overcome these technical difficulties, we redefine the sample path ω as $\omega = \{(l_t, q_t, \zeta_t)\colon t = 1, \ldots, T\}$, where each $\zeta_{t,\,i}$ is an i.i.d. random variable uniformly distributed on $[0, \epsilon]$, for some small ϵ. Although ζ_t is a function of ω, we do not explicity express this dependence to simplify the notation. Then, we consider the following variation of the problem:

$$R_t(x(t), y, \omega) = r^{\mathsf{T}} \cdot u(x(t) - \zeta_t, y, l_t, q_t) + R_{t+1}(x(t+1), y, \omega) \tag{A1}$$

$$x(t+1) = x(t) - \zeta_t - A \cdot u(x(t) - \zeta_t, y, l_t, q_t); \tag{A2}$$

The purpose is to smooth the acceptance function by randomly perturbing the remaining capacity. With this new formulation, following with the argument in the previous paragraph, the event $y_{i,\,c_i(j)-1} = x_i(t) - \zeta_{t,\,i}$ occurs with probability zero. Therefore, the control $u(\cdot)$ defined in (A2) becomes differentiable w.p.1. Using the composition defined by (A1), it is not hard to see that the revenue function becomes differentiable w.p.1. as well.

Using the chain rule, we then obtain the set of *backward equations* for the derivatives with respect to y_{ic}:

$$\begin{aligned}\frac{\partial}{\partial y_{ic}} R_t(x(t), y, \omega) &= (r^{\mathsf{T}} - \nabla_{x(t+1)} R_{t+1}(x(t+1), y, \omega) \cdot A) \cdot \frac{\partial}{\partial y_{ic}} u(x(t) - \zeta_t, y, l_t, q_t) \\ &\quad + \frac{\partial}{\partial y_{ic}} R_{t+1}(x(t+1), y, \omega), \quad \forall\, i, c, t.\end{aligned} \tag{A3}$$

We get a similar set of backward equations for the derivatives with respect to x_i:

$$\begin{aligned}\frac{\partial}{\partial x_i(t)} R_t(x(t), y, \omega) &= (r^{\mathsf{T}} - \nabla_{x(t+1)} R_{t+1}(x(t+1), y, \omega) \cdot A) \\ &\quad \cdot \frac{\partial}{\partial x_i(t)} u(x(t) - \zeta_t, y, l_t, q_t) \\ &\quad + \frac{\partial}{\partial x_i(t+1)} R_{t+1}(x(t+1), y, \omega), \quad \forall\, i, t;\end{aligned} \tag{A4}$$

with boundary conditions

$$\frac{\partial}{\partial y_{ic}} R_{T+1}(x(T+1), y, \omega) = 0, \quad \forall\, i, c;$$

$$\frac{\partial}{\partial x_i(t+1)} R_{T+1}(x(T+1), y, \omega) = 0, \quad \forall\, i.$$

(In the next subsection, we include a detailed derivation of the partial derivatives for the revenue function.)

Note that the general form of the two gradients is very similar. The term in the parentheses is simply the marginal revenue for accepting one extra unit of product j minus the marginal displacement cost over the legs used by product j—in other words, product j's displacement-adjusted revenue value. This quantity is multiplied by the gradients of the acceptance function $((\partial/\partial y_{ic})u_j(x, y, l, q)$ or $(\partial/\partial x_i)u_j(x, y, l, q))$ to give the marginal value in the current period. Adding this to the marginal revenue-to-go gives the total gradient.

Appendix B. Derivation of the Partial Derivatives for the Revenue Function

From Equation (A1), the derivation of the partial derivative of the revenue function with respect to y_{ic} proceeds as follows:

$$\begin{aligned}
&\frac{\partial}{\partial y_{ic}} R_t(x(t), y, \omega) \\
&\quad = r^{\top} \cdot \frac{\partial}{\partial y_{ic}} u(x(t) - \zeta_t, y, l_t, q_t) \\
&\qquad + \nabla_{x(t+1)} R_{t+1}(x(t+1), y, \omega) \cdot \frac{\partial}{\partial y_{ic}} x(t+1) \\
&\qquad + \frac{\partial}{\partial y_{ic}} R_{t+1}(x(t+1), y, \omega),
\end{aligned}$$

where

$$\frac{\partial}{\partial y_{ic}} x(t+1) = -A \cdot \frac{\partial}{\partial y_{ic}} u(x(t) - \zeta_t, y, l_t, q_t).$$

Then,

$$\begin{aligned}
&\frac{\partial}{\partial y_{ic}} R_t(x(t), y, \omega) \\
&= (r^{\top} - \nabla_{x(t+1)} R_{t+1}(x(t+1), y, \omega) \times A) \cdot \frac{\partial}{\partial y_{ic}} u(x(t) - \zeta_t, y, l_t, q_t) \\
&\quad + \frac{\partial}{\partial y_{ic}} R_{t+1}(x(t+1), y, \omega). \qquad \text{(B1)}
\end{aligned}$$

Now we have to solve for the partial derivative with respect to capacity. Taking again Equation (A1):

$$\begin{aligned}
&\frac{\partial}{\partial x_i(t)} R_t(x(t), y, \omega) \\
&\quad = r^{\top} \cdot \frac{\partial}{\partial x_i(t)} u(x(t) - \zeta_t, y, l_t, q_t) \\
&\qquad + \nabla_{x(t+1)} R_{t+1}(x(t+1), y, \omega) \cdot \frac{\partial}{\partial x_i(t+1)} x(t+1).
\end{aligned}$$

The partial derivative of the remaining capacity function is

$$\frac{\partial}{\partial x_i(t+1)} x(t+1) = e_i - A \cdot \frac{\partial}{\partial x_i(t)} u(x(t) - \zeta_t, y, l_t, q_t).$$

Regrouping terms, we have that

$$\begin{aligned}
&\frac{\partial}{\partial x_i(t)} R_t(x(t), y, \omega) \\
&\quad = (r^{\top} - \nabla_{x(t+1)} R_{t+1}(x(t+1), y, \omega) \times A) \\
&\qquad \cdot \frac{\partial}{\partial x_i(t)} u(x(t) - \zeta_t, y, l_t, q_t) \\
&\qquad + \frac{\partial}{\partial x_i(t+1)} R_{t+1}(x(t+1), y, \omega). \qquad \text{(B2)}
\end{aligned}$$

Appendix C. Gradients of u_j

We next determine the gradients of $u_j(x, y, l_t, q_t)$. To ensure that partial derivatives are well defined on the boundary of the feasible set (1), we redefine (2) as follows.

$$x_i^j(t) = \min\left\{\left(x_i(t) - y_{i,c} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x(t), y, l_t, q_t)\mathbf{I}\{i \in A_{[k]}\}]\right)^+ : i \in A_j,\ c < c_i(j)\right\} \qquad \text{(C1)}$$

Assuming there is a unique minimum in definition (3), then from (C1), one can determine for all i and c the following partial derivative.

$$\frac{\partial}{\partial y_{ic}} u_j(x, y, l_t, q_t)$$

$$= \begin{cases}
-1 & \text{if protection level } y_{ic} \text{ is uniquely binding for product } j, \text{ i.e., if simultaneously} \\
& \text{(i) } i \in A_j \\
& \text{(ii) } x_i - y_{i,c} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x, y, l_t, q_t)\mathbf{I}\{i \in A_{[k]}\}] \\
& \qquad < x_h - y_{h,c'} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x, y, l_t, q_t)\mathbf{I}\{h \in A_{[k]}\}], \\
& \qquad \forall h \in A_j, \text{ and } \forall c' < c_h(j), \text{ with } (i,c) \neq (h,c') \\
& \text{(iii) } 0 < x_i - y_{i,c} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x, y, l_t, q_t)\mathbf{I}\{i \in A_{[k]}\}] \\
& \qquad < q_t - \sum_{k=1}^{b(t,j)-1} u_{[k]}(x, y, l_t, q_t) \\
& \text{(iv) } c < c_i(j) \\
-\sum_{k=1}^{b(t,j)-1} & \left[\frac{\partial}{\partial y_{ic}} u_{[k]}(x, y, l_t, q_t)\mathbf{I}\{l \in A_{[k]}\}\right] \\
& \text{if for some } l,\ l \in A_j, \text{ there exists a } d < c_l(j), \text{ such that the protection level } y_{ld}, \text{ with } (l,d) \neq (i,c),
\end{cases}$$

$$
\begin{cases}
\text{is uniquely binding for } j\text{, i.e., the following hold:} \\
\text{(i) } x_l - y_{ld} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{l \in A_{[k]}\}] \\
\qquad < x_h - y_{h,c'} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{h \in A_{[k]}\}], \\
\qquad\quad \forall\, h \in A_j, \text{ and } \forall\, c' < c_h(j), \text{ with } (l,d) \neq (h,c') \\
\text{(ii) } 0 < x_l - y_{ld} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{l \in A_{[k]}\}] \\
\qquad < q_t - \sum_{k=1}^{b(t,j)-1} u_{[k]}(x,y,l_t,q_t) \\
-\sum_{k=1}^{b(t,j)-1} \dfrac{\partial}{\partial y_{ic}} u_{[k]}(x,y,l_t,q_t)
\end{cases}
$$

if there is a positive amount allocated to j, but no constrained by a binding protection level, i.e.,

$$
0 < u_j(x,y,l_t,q_t) = q_t - \sum_{k=1}^{b(t,j)-1} u_{[k]}(x,y,l_t,q_t) < x_h^j, \quad \forall\, h \in A_j
$$

$$
0 \quad \text{if } u_j(x,y,l_t,q_t) = 0
$$

(C2)

If there is no unique minimum in definition (3), then the derivative does not exist.

In the first case of (C2), the quantity of demand accepted of product j for customer t in state x is reduced (one-for-one) by a slight increase in the protection level y_{ic} if and only if all of the following hold: (i) resource i is used by j, (ii) the capacity available for product j on resource i is a binding constraint, (iii) the amount accepted is positive but constrained by the protection level associated to product j over resource i, and (iv) class c is higher in the nesting order than the virtual class of product j over resource i. Note that the complete form of this first case for the derivative in Equation (C2) should be:

$$
\frac{\partial}{\partial y_{ic}} u_j(x,y,l_t,q_t) = -1 - \sum_{k=1}^{b(t,j)-1} \left[\frac{\partial}{\partial y_{ic}} u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{i \in A_{[k]}\} \right].
$$

However, each term in the sum is null: if protection level y_{ic} is binding at time t for product j, and j is allocated a positive amount y_{ie}, it could not have been binding for products higher than j in the preference order, for which also the quantity q_t had not been exhausted yet. So, the partial derivative is just -1.

Cases 2 and 3 in (C2) take care of cross-network effects. The second case occurs when the amount of product j accepted is positive and determined by a uniquely binding protection level y_{ld} where $(l,d) \neq (i,c)$. Here, if we marginally increase the protection level y_{ic}, which could be binding for a product that uses leg $l \in A_j$ but that is located higher than j in the preference list, the marginal quantity spilled from above could be received by product j. The third case occurs when the quantity q_t is exhausted at product j and there is no binding protection level. Here, product j will receive any spillover from above.

A similar reasoning provides the derivatives with respect to x_i: Assuming there is a unique minimum in definition (3), then from (C1), one can determine for each resource i the following partial derivative.

$$
\frac{\partial}{\partial x_i} u_j(x,y,l_t,q_t) =
\begin{cases}
1 \quad \text{if resource } i \text{ is uniquely binding for product } j, \\
\quad \text{i.e., if } i \in A_j \text{, and for some } c < c_i(j), \\
\quad \text{the following hold:} \\
\quad \text{(i) } x_i - y_{i,c} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{i \in A_{[k]}\}] \\
\qquad < x_h - y_{h,c'} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{h \in A_{[k]}\}], \\
\qquad\quad \forall\, h \in A_j, \text{ and } \forall\, c' < c_h(j), \text{ with } (i,c) \neq (h,c') \\
\quad \text{(ii) } 0 < x_i - y_{i,c} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{i \in A_{[k]}\}] \\
\qquad < q_t - \sum_{k=1}^{b(t,j)-1} u_{[k]}(x,y,l_t,q_t) \\
-\sum_{k=1}^{b(t,j)-1} \left[\dfrac{\partial}{\partial x_i} u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{l \in A_{[k]}\} \right] \\
\quad \text{if for some } l,\ l \in A_j,\ l \neq i, \text{ there exists a } d < c_l(j), \\
\quad \text{such that the protection level } y_{ld} \text{ is uniquely} \\
\quad \text{binding for } j\text{, i.e., the following hold:} \\
\quad \text{(i) } x_l - y_{ld} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{l \in A_{[k]}\}] \\
\qquad < x_h - y_{h,c'} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{h \in A_{[k]}\}], \\
\qquad\quad \forall\, h \in A_j, \text{and } \forall\, c' < c_h(j), \text{ with } (l,d) \neq (h,c') \\
\quad \text{(ii) } 0 < x_l - y_{ld} - \sum_{k=1}^{b(t,j)-1} [u_{[k]}(x,y,l_t,q_t)\mathbf{I}\{l \in A_{[k]}\}] \\
\qquad < q_t - \sum_{k=1}^{b(t,j)-1} u_{[k]}(x,y,l_t,q_t)
\end{cases}
$$

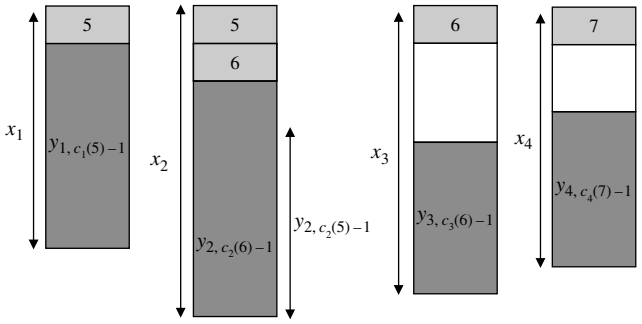
$$\begin{cases} -\sum_{k=1}^{b(t,j)-1} \dfrac{\partial}{\partial x_i} u_{[k]}(x, y, l_t, q_t) \\ \quad \text{if there is a positive amount allocated to } j, \\ \quad \text{but no constrained by a binding protection level, i.e.,} \\ \quad 0 < u_j(x, y, l_t, q_t) = q_t - \sum_{k=1}^{b(t,j)-1} u_{[k]}(x, y, l_t, q_t) \\ \quad\quad < x_h^j, \quad \forall h \in A_j \\ 0 \quad \text{if } u_j(x, y, l_t, q_t) = 0 \end{cases} \tag{C3}$$

If there is no unique minimum in definition (3), then the derivative does not exist. The interpretation is analogous to the derivative with respect to y_{ic}.

Conditions for the partial derivatives of the acceptance function are further illustrated in Figure A.1. The height of the bars represents the capacity remaining at time t, and the quantities $y_{i,c_i(j)-1}$ represent the protection levels for product j on each resource i. Customer t requests $q_t = 3$ seats. His first choice is product 5, which uses legs 1 and 2; his second choice is product 6, which uses legs 2 and 3, and the last choice is product 7, which uses leg 4. Following the notation we introduced in §2: $l_{t1} = 5$, $l_{t2} = 6$, $l_{t3} = 7$, and $p_t = 3$. Note in Figure A.1 that leg 1 is binding for product 5, and leg 2 is binding for product 6. These binding constraints forces the allocation of the third seat to product 7. Some straight forward partial derivatives are:

$$\frac{\partial}{\partial y_{1,c_1(5)-1}} u_5(x, y, l_t, 3) = -1, \quad \frac{\partial}{\partial y_{2,c_2(5)-1}} u_5(x, y, l_t, 3) = 0,$$
$$\frac{\partial}{\partial y_{2,c_2(6)-1}} u_6(x, y, l_t, 3) = -1, \quad \frac{\partial}{\partial y_{3,c_3(6)-1}} u_6(x, y, l_t, 3) = 0,$$
$$\frac{\partial}{\partial y_{4,c_4(7)-1}} u_7(x, y, l_t, 3) = 0.$$

Figure A.1 **Acceptance Function for Customer t, Requesting $q_t = 3$ with Preferences $l_t = [5, 6, 7, 0, \ldots, 0]$**



Note. Product 5 uses legs 1 and 2; product 6 uses legs 2 and 3, and product 7 uses leg 4.

The cross network effects make some derivatives not so straightforward. For example, it can be verified that

$$\frac{\partial}{\partial y_{1,c_1(5)-1}} u_6(x, y, l_t, 3) = 1,$$

meaning that when slightly incrementing the protection level $y_{1,c_1(5)-1}$, the reduction in the number of product 5 accepted is compensated by an increase in the number of product 6 accepted. This is captured by case 2 in formula (C2). Another cross-network effect is given by case 3 in formula (C2), from where it can be verified that

$$\frac{\partial}{\partial y_{1,c_1(5)-1}} u_7(x, y, l_t, 3) = 0.$$

Case 3 also leads to the partial derivative

$$\frac{\partial}{\partial y_{2,c_2(6)-1}} u_7(x, y, l_t, 3) = 1.$$

Analogously, we can compute the partial derivatives with respect to capacity. For instance, from formula (C3), it can be verified that

$$\frac{\partial}{\partial x_1} u_5(x, y, l_t, 3) = 1, \quad \frac{\partial}{\partial x_2} u_5(x, y, l_t, 3) = 0,$$
$$\frac{\partial}{\partial x_2} u_6(x, y, l_t, 3) = 1, \quad \frac{\partial}{\partial x_3} u_6(x, y, l_t, 3) = 0,$$
$$\frac{\partial}{\partial x_4} u_7(x, y, l_t, 3) = 0.$$

Appendix D. Example Calculation of Revenue Function Gradients

We will analyze the gradients of the revenue function for three toy examples.

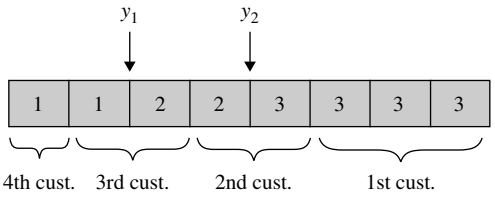
Example 1. Take a single leg problem, with 3 products, 3 virtual classes (one per product) and revenues $r = (25, 19, 10)$. Suppose $x = 8$, and protection levels are $y = (2, 4)$. Consider a sample path ω with four requests:

$$\omega = (([3,2,0],3),([3,2,0],2),([2,1,0],2),([2,1,0],2)).$$

The sample path is processed as follows: the first booking is fulfilled using product 3. The second request is assigned one seat of product 3, but then product 3 meets a binding constraint (e.g., after the realization of the random variable ζ, $x(2) - \zeta = 4.993$) because 4 seats out of 8 are reserved for classes 1 and 2. Hence, the second seat is allocated to product 2. Something similar occurs with the third customer: her first seat is sold on product 2, but the protection level y_1 is hit (e.g., $x(3) - \zeta = 2.986$, and then the second seat is allocated to product 1. There is just one seat left for the fourth customer, and a product 1 is sold to him. Figure A.2 illustrates the selling process, where seats are filled from right to left.

In this case, when marginally decreasing capacity x, we will be marginally decreasing the quantity of product 3

Figure A.2 Selling Process for Example 1 in Appendix D



Note. The numbers in the grey boxes indicate the product allocated to the corresponding customer.

accepted. Missing a marginal unit of product 3 translates into:

$$\nabla_x R_1(x, y, \omega) = 10.$$

Regarding the partial derivatives with respect to the protection levels, note that by marginally incrementing y_2, we will reject a marginal unit of product 3, but will accept a marginal unit of product 2, changing the revenue in $r_2 - r_3 = 9$. Likewise, by marginally incrementing y_1, we will accept an additional increment of product 1 for the third customer (at the expense of a marginal unit of product 2), leading to a revenue change of $r_1 - r_2 = 6$. Hence,

$$\nabla_y R_1(x, y, \omega) = (6, 9).$$

Example 2. Again, consider the same single leg problem: 3 products, 3 virtual classes (one per product) and revenues $r = (25, 19, 10)$, with capacity $x = 8$, and protection levels $y = (2, 4)$. Now, take the following sample path ω with three requests:

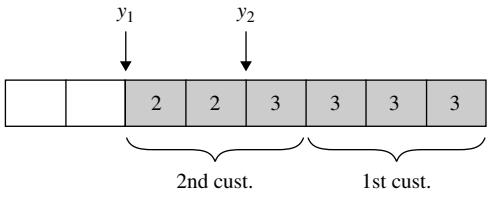
$$\omega = (([3, 0, 0], 3), ([3, 2, 1], 3), ([2, 0, 0], 2)).$$

The selling process is represented in Figure A.3. In a nonperturbed framework, the first customer gets three products 3. The second customer is assigned one product 3, and two products 2. There is no availability for the last customer.

Here, when processing the second customer and perturbing capacity (e.g., $x(2) - \zeta = 4.989$), a marginal unit of product 3 is compensated with an increment of product 1, changing the revenue by $r_3 - r_1 = -15$.

$$\nabla_x R_1(x, y, \omega) = -15.$$

Figure A.3 Selling Process for Example 2 in Appendix D



Note. The numbers in the grey boxes indicate the product allocated to the corresponding customer.

Observe that this translates into an increase in the sample path revenue. The main point here is that by taking advantage of the substitution effect, it could be worthwhile for the seller to introduce some scarcity in the availability of lower classes to improve revenue performance.

Fixing $x = 8$, when studying the derivative with respect to y_2, request from the second customer will be fully processed with an additional increment of product 2, allowing for an increase of $r_2 - r_3 = 9$. Regarding the derivative with respect to y_1 (fixing y_2), the second customer will get a marginal unit of product 1 at the expense of a marginal unit of product 2, leading to an increase of $r_1 - r_2 = 6$. The gradient of the revenue with respect to y is then

$$\nabla_y R_1(x, y, \omega) = (6, 9).$$

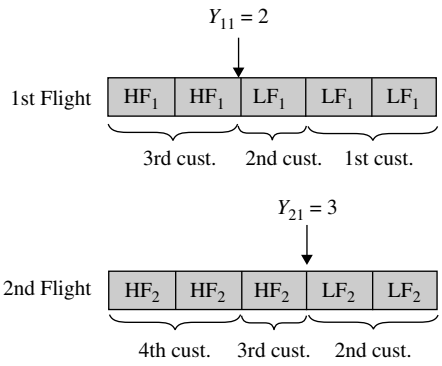
Example 3. Assume that there are two alternatives covering an origin-destination pair (e.g., two direct flights between cities A and B: first one is at 7 a.m.; the second is at 10 a.m.). Consider the same capacity for both: $x_1 = x_2 = 5$. There are two products (classes) per flight, which for simplicity are denoted: LF_k (low fare class for flight k) and HF_k (high fare class for flight k), with $k = 1, 2$. The protection level for HF_1 is $y_{11} = 2$; and the protection level for HF_2 is $y_{21} = 3$. The revenues are 150 for HF_1, 170 for HF_2, and 100 for both LF_1 and LF_2.

Take the following sample path ω with four requests (following the notation of Example 1):

$$\omega = (([\mathrm{LF}_1, \mathrm{LF}_2, 0, 0], 2), ([\mathrm{LF}_2, \mathrm{LF}_1, 0, 0], 3),$$
$$([\mathrm{HF}_1, \mathrm{HF}_2, 0, 0], 3), ([\mathrm{HF}_2, \mathrm{HF}_1, 0, 0], 3)).$$

The selling process in a nonperturbed setting is represented in Figure A.4. The first customer gets her first choice. The second customer gets two seats in the second flight, and one in the early morning flight. The third customer gets two seats in the first flight, and one in the second one. The last customer can be allocated just two seats in the second flight.

Figure A.4 Selling Process for Example 3 in Appendix D



Note. The labels in the grey boxes indicate the product allocated to the customer in the brackets below.

Suppose we slightly increase the value of y_{11}. In this case, the third unit of the second customer will be marginally decreased, but both customers 3 and 4 will be marginally more satisfied with their first choice. That is, in terms of marginal units, we lose one LF_1, we accommodate one current of HF_2 as HF_1, and we assign one extra HF_2 for customer 4.

$$\frac{\partial}{\partial y_{11}} R_1(x, y, \omega) = 170 + (150 - 170) - 100 = 50.$$

If we slightly increase the value of y_{21}, we basically reject one marginal unit for the second customer, and we are able to meet an additional demand unit of the fourth customer:

$$\frac{\partial}{\partial y_{21}} R_1(x, y, \omega) = 170 - 100 = 70.$$

Regarding the partial with respect to capacities: If we decrease either x_1 or x_2 by one marginal unit, we lose one marginal seat for the second customer, leading to

$$\frac{\partial}{\partial x_1} R_1(x, y, \omega) = \frac{\partial}{\partial x_2} R_1(x, y, \omega) = 100.$$

References

Algers, S., M. Besser. 2001. Modeling choice of flight and booking class: A study using stated preference and revealed preference data. *Internat. J. Services Tech. Management* **2** 28–45.

Andersson, S. E. 1989. Operational planning in airline business—Can science improve efficiency? Experiences from SAS. *Eur. J. Oper. Res.* **43** 3–12.

Andersson, S. E. 1998. Passenger choice analysis for seat capacity control: A pilot project in Scandinavian Airlines. *Internat. Trans. Oper. Res.* **5** 471–486.

Belobaba, P. 1987. Air travel demand and airline seat inventory management. Ph.D. thesis, Flight Transportation Laboratory, MIT, Cambridge, MA.

Belobaba, P., C. Hopperstad. 1999. Boeing/MIT simulation study: PODS results update. *Proc. 1999 AGIFORS Reservations Yield Management Study Group Sympos.* London, UK.

Belobaba, P. P., L. R. Weatherford. 1996. Comparing decision rules that incorporate customer diversion in perishable asset revenue management situations. *Decision Sci.* **27** 343–363.

Benveniste, A., M. Métivier, P. Priouret. 1990. *Adaptive Algorithms and Stochastic Approximations*. Springer-Verlag, Berlin.

Bertsekas, D. 1999. *Nonlinear Programming*, 2nd ed. Athena Scientific, Nashua, NH.

Bertsimas, D., S. de Boer. 2005. Simulation-based booking-limits for airline revenue management. *Oper. Res.* **53** 90–106.

Boyd, E. A., R. Kallesen. 2004. The science of revenue management when passengers purchase the lowest available fare. *J. Revenue Pricing Management* **3** 171–177.

Gallego, G., G. Iyengar, R. Phillips, A. Dubey. 2004. Managing flexible products on a network. Technical Report CORC TR-2004-01, Department of Industrial Engineering and Operations Research, Columbia University, New York.

Glasserman, P. 1994. Perturbation analysis of production networks. D. D. Yao, ed. *Stochastic Modeling and Analysis of Manufacturing Systems*, Chapter 6. Springer, New York, 233–280.

Kleywegt, A. J., A. Shapiro. 2001. Stochastic optimization. G. Salvendy, ed. *Handbook of Industrial Engineering*, 3rd ed. John Wiley and Sons, New York, 2625–2650.

Kreps, D. M. 1988. *Notes on the Theory of Choice*. Westview Press, London.

Kushner, H., G. Yin. 2003. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer-Verlag, New York.

Kushner, H. J., D. S. Clark. 1978. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. Springer-Verlag, Berlin.

Littlewood, K. 1972. Forecasting and control of passengers bookings. *12th AGIFORS Sympos. Proc.* Nathanya, Israel, 95–128.

Liu, Q., G. J. van Ryzin. 2008. On the choice-based linear programming model for network revenue management. *Manufacturing Service Operations Management* **10**(2) 288–310.

Mahajan, S., G. J. van Ryzin. 2001. Stocking retail assortments under dynamic consumer substitution. *Oper. Res.* **49** 334–351.

Pflug, G. 1988. Step size rules, stopping times and their implementation in stochastic quasigradient methods. Y. Ermoliev, R. J.-B. Wets, eds. *Numerical Techniques in Stochastic Optimization*, Chap. 17. Springer-Verlag, Berlin, 353–372.

Phillips, R. L. 1994. State-contingent airline yield management. Presentation, Session TC33.4, INFORMS, Detroit.

Robbins, H., S. Monro. 1951. On a stochastic approximation method. *Ann. Math. Statist.* **22** 400–407.

Shapiro, A. 2000. Stochastic programming by Monte Carlo simulation methods. *Stochastic Programming E-Prints Series*, Article 2000-03.

Smith, B., J. Leimkuhler, R. Darrow. 1992. Yield management at American airlines. *Interfaces* **22** 8–31.

Talluri, K. T., G. J. van Ryzin. 2004a. Revenue management under a general discrete choice model of consumer behavior. *Management Sci.* **50**(January) 15–33.

Talluri, K. T., G. J. van Ryzin. 2004b. *The Theory and Practice of Revenue Management*. Kluwer Academic Press, New York.

van Ryzin, G. J., G. Vulcano. 2006. Simulation-based optimization of virtual nesting controls for network revenue management. *Oper. Res.* Forthcoming.

Vulcano, G., G. J. van Ryzin. 2008. Choice-based revenue management: An empirical study of estimation and optimization. Working paper, Stern School of Business, New York University.

Williamson, E. 1992. Airline network seat inventory control: Methodologies and revenue impacts. Ph.D. thesis, Flight Transportation Laboratory, MIT, Cambridge, MA.

Zhang, D., W. L. Cooper. 2005. Revenue management for parallel flights with customer choice behavior. *Oper. Res.* **53** 414–431.