

# Randomized Algorithms for Lexicographic Inference

Rajeev Kohli,<sup>a</sup> Khaled Boughanmi,<sup>a</sup> Vikram Kohli<sup>b</sup>

<sup>a</sup> Graduate School of Business, Columbia University, New York, New York 10027; <sup>b</sup> McCormick School of Engineering, Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, Illinois 60208

Contact: rk35@columbia.edu,  <http://orcid.org/0000-0003-3466-8028> (RK); kboughanmi18@gsb.columbia.edu (KB); vikram.kohli@northwestern.edu (VK)

Received: January 7, 2017

Revised: November 2, 2017; April 4, 2018

Accepted: June 13, 2018

Published Online in Articles in Advance:  
March 18, 2019

## Subject Classifications:

Marketing: choice models, estimation/statistical techniques; analysis of algorithms: computational complexity, suboptimal algorithms; mathematics: combinatorics; programming: heuristics; utility/preference: applications, choice functions, estimation, multiattribute, scaling

Area of Review: Revenue Management and Market Analytics

**Abstract.** The inference of a lexicographic rule from paired comparisons, ranking, or choice data is a discrete optimization problem that generalizes the linear ordering problem. We develop an approach to its solution using randomized algorithms. First, we show that maximizing the expected value of a randomized solution is equivalent to solving the lexicographic inference problem. As a result, the discrete problem is transformed into a continuous and unconstrained nonlinear program that can be solved, possibly only to a local optimum, using nonlinear optimization methods. Second, we show that a maximum likelihood procedure, which runs in polynomial time, can be used to implement the randomized algorithm. The maximum likelihood value determines a lower bound on the performance ratio of the randomized algorithm. We employ the proposed approach to infer lexicographic rules for individuals using data from a choice experiment for electronic tablets. These rules obtain substantially better fit and predictions than a previously described greedy algorithm, a local search algorithm, and a multinomial logit model.

<https://doi.org/10.1287/opre.2018.1794>

Copyright: © 2019 INFORMS

**Keywords:** lexicographic rules • noncompensatory choice • consumer search • discrete optimization • randomized algorithms • analysis of algorithms • maximum likelihood

## 1. Introduction

Online retailers often design web pages to facilitate product search. Figure 1 shows the example of [Amazon.com](https://www.amazon.com), which on a recent day had more than 6,000 different electronic tablets available for sale. The menu on the left allows a shopper to search for alternatives using one or more criteria: flash size, display size, weight, etc. The set of matching alternatives, shown on the right side in Figure 1, is reduced each time a criterion is added. This sequential screening represents a lexicographic rule. Often, it uses only a few steps to identify attractive options among hundreds or thousands of alternatives. Research in marketing and psychology suggests that people use lexicographic rules in several situations (see Section 2), one of which is illustrated by the preceding example: there are many options, and a decision maker has insufficient time, information, and/or ability to make trade-off comparisons across all the alternatives.

*Lexicographic inference* is the problem of deducing a lexicographic rule from consumer preferences or choices over alternatives, each of which is described using discrete (or discretized) attributes. For example, in Figure 1, each attribute, such as brand name or price, has a small number of discrete “levels” (values). Kohli and Jedidi (2007) and Yee et al. (2007) described methods for solving the lexicographic inference problem using













paired comparisons data. These data may be directly obtained or inferred from individual choices or rankings of alternatives. The solution to the problem is a lexicographic ordering of the attribute levels consistent with the maximum number of paired comparisons. Yee et al. (2007) described an (exponential time) dynamic program, which can be used for solving small problems; for larger problems, they and Kohli and Jedidi (2007) proposed a greedy heuristic. Schmitt and Martignon (2006) considered a generalization of the lexicographic inference problem in which alternatives can have different numbers of attribute levels. They showed that, unless  $P=NP$ , no polynomial-time algorithm can guarantee a solution that exceeds any constant fraction of the optimal solution value. However, this result extends only partly to the problem considered by Yee et al. (2007) and Kohli and Jedidi (2007), in which each alternative has the same number of levels, one level per attribute. As we show, in this case the problem remains NP-hard (it generalizes the linear ordering problem) but allows polynomial-time approximation algorithms with nontrivial performance bounds.

We consider randomized algorithms for solving the lexicographic inference problem. Unlike the deterministic algorithms just described, a randomized algorithm introduces uncertainty in assigning a value to a decision

**Figure 1.** (Color online) Online Retail Store Featuring Multiple Criteria (Brand, Price, Operating System, etc.) for Screening Electronic Tablets

Refine By	Featured	All
Flash Size	Showing featured results. Click All to see all results.	
Display Size		
Weight		
Connectivity Technology		
Operating System		
Customer Reviews		
Condition		
Price		

 Fire Tablet, 7" Display, Wi-Fi, 8 GB - Includes Special Offers, Black by Amazon \$39.99 Prime ★★★★★ - 89155	 Fire Tablet, 7" Display, Wi-Fi, 8 GB - Includes Special Offers, Blue by Amazon \$39.99 Prime ★★★★★ - 89156	 All-New Fire HD 8 Tablet, 8" HD Display, Wi-Fi, 16 GB - Includes Special Offers, Black by Amazon \$89.99 Prime ★★★★★ - 13630	 All-New Fire HD 8 Tablet, 8" HD Display, Wi-Fi, 16 GB - Includes Special Offers, Blue by Amazon \$89.99 Prime ★★★★★ - 13631
 Fire Tablet, 7" Display, Wi-Fi, 8 GB - Includes Special Offers, Magenta by Amazon \$39.99 Prime ★★★★★ - 89155	 All-New Fire HD 8 Tablet, 8" HD Display, Wi-Fi, 32 GB - Includes Special Offers, Black by Amazon \$99.99 Prime ★★★★★ - 13630	 All-New Fire HD 8 Tablet, 8" HD Display, Wi-Fi, 16 GB - Includes Special Offers, Magenta by Amazon \$89.99 Prime ★★★★★ - 13630	 Fire Tablet, 7" Display, Wi-Fi, 8 GB - Includes Special Offers, Tangerine by Amazon \$39.99 Prime ★★★★★ - 89155
 Fire Kids Edition Tablet, 7" Display, Wi-Fi, 16 GB, Pink Kid-Proof Case by Amazon \$89.99 Prime ★★★★★ - 13739	 Fire Tablet, 7" Display, Wi-Fi, 16 GB - Includes Special Offers, Black by Amazon \$49.99 Prime ★★★★★ - 89156	 Fire Kids Edition Tablet, 7" Display, Wi-Fi, 16 GB, Blue Kid-Proof Case by Amazon \$89.99 Prime ★★★★★ - 13739	 Fire Kids Edition Tablet, 7" Display, Wi-Fi, 16 GB, Green Kid-Proof Case by Amazon \$89.99 Prime ★★★★★ - 13739

variable. A randomized algorithm can be a Las Vegas algorithm, which obtains an optimal solution but has uncertain running time, or a Monte Carlo algorithm, which has a deterministic running time but an uncertain solution value. We consider Monte Carlo algorithms in this paper, using the data for a problem instance to determine the probability with which it selects a solution. The performance of such an algorithm can be assessed by comparing the expected value of its solution to the optimal solution value.

The proposed randomized algorithm is based on a random utility formulation of the lexicographic inference problem that was shown by Kohli and Jedidi (2015) to be equivalent to Tversky's (1972) elimination by aspects. Let  $u_j = v_j + \epsilon_j$  denote the utility of attribute level  $j$  (for example, the utility associated with a particular brand name or price level), where  $v_j$  is a deterministic utility component and  $\epsilon_j$  is a stochastic utility component with an independent extreme value distribution. Suppose we knew the  $v_j$  values. Then we could use the following method to obtain a solution:

generate independent  $\epsilon_j$  values, calculate  $u_j$ , and obtain a feasible ordering of the attribute levels by arranging them in decreasing order of their  $u_j$  values. The key step is determining the  $v_j$  values using the data for a problem instance. We consider two methods: (1) maximizing the expected value of the solution obtained by a randomized algorithm and (2) maximizing a likelihood function. To our knowledge, these are both new approaches to designing randomized algorithms. We obtain the following results:

(1) We show that maximizing the expected value of the solution obtained by the randomized algorithm is equivalent to optimally solving the lexicographic inference problem. Because the problem is NP-hard, we cannot maximize the expected value in polynomial time unless  $P=NP$ . However, the expected value formulation is useful because it transforms the discrete problem into an unconstrained and continuous nonlinear optimization problem. This allows using continuous optimization procedures to obtain at least locally optimal solutions. To our knowledge, maximizing an expected value is a new

way of formulating a discrete optimization problem as a continuous nonlinear optimization problem (see Floudas and Visweswaran 1995 for other methods that can be used to obtain such reformulations).

(2) We propose a randomized algorithm in which the  $v_j$  values are obtained by maximizing a likelihood function. This is a convex optimization problem that can be solved in polynomial time. We use the maximum likelihood values of  $v_j$  to implement a Monte Carlo algorithm and select the best solution across runs (which, in turn, can be used as a starting solution to the nonlinear optimization problem described in (1)).

We obtain a relation between the maximum likelihood solution and the optimal solution to the lexicographic inference problem. The relation is based on the observation that maximizing a likelihood function is equivalent to maximizing a geometric mean of probabilities, and maximizing an expected value is equivalent to maximizing an arithmetic mean of the same probabilities. The relation between arithmetic and geometric means provides a lower bound on the expected value of the randomized solution in terms of the maximum likelihood value. This bound can be computed for each problem instance.

We present an application to illustrate the proposed methods using data from a choice experiment for tablet computers. We examine the performance of aggregate and individual level lexicographic rules inferred by (1) a randomized algorithm using the maximum likelihood approach and (2) maximizing the expected value of the randomized algorithm. We compare these solutions to those obtained by using (1) the greedy algorithm described by Yee et al. (2007) and Kohli and Jedidi (2007), (2) a local search algorithm, (3) the ordering of the maximum likelihood values of  $v_j$ , and (4) a logit model. The proposed algorithms perform substantially better than these methods at both aggregate and individual levels.

## 1.1 Organization of the Paper

Section 2 provides a background on lexicographic rules, the lexicographic inference problem, the linear ordering problem, and randomized algorithms. Section 3 first obtains a discrete formulation of the lexicographic inference problem and shows that it is NP-hard. Then it describes the continuous formulation obtained by maximizing the expected value of the randomized algorithm. Section 4 describes the maximum likelihood approach and characterizes the worst-case performance of the randomized algorithm. Section 5 describes the empirical application.

## 2. Background

### 2.1. Lexicographic Preferences

We consider discrete (or discretized) attributes. Regardless of whether it is nominal or ordered, we

refer to each discrete value of an attribute as a level. A lexicographic rule orders alternatives over attribute levels in the same way that a dictionary orders words over letters. It is a noncompensatory rule: one alternative is preferred to another if it is better on the most important attribute level on which two alternatives are different.

A substantial literature in psychology and marketing documents the use of lexicographic rules. Consumers use them when it is costly to retrieve information about alternatives (Bröder 2000, Bröder and Schiffer 2003), when they have to make choices or directly compare alternatives (Payne 1982, Billings and Scherer 1988, Tversky et al. 1988, Schkade and Johnson 1989, Bettman et al. 1998), when they have emotional reasons to avoid trade-offs (Drolet and Luce 2004), and when they need to break ties among equally valued alternatives (Slovic 1975). Kohli and Jedidi (2007) and Yee et al. (2007) reported that approximately two thirds of their subjects used lexicographic rules for ranking personal computers and smartphones, respectively. Sometimes, people use a lexicographic rule when making judgments based on cues. For example, they may judge a city to be larger if it has a professional soccer team; see Bröder (2000), Gigerenzer et al. (1991), and Dieckmann et al. (2009).

Parallel to this literature, theoretical research has examined the representation of lexicographic preferences by utility functions. It has been known at least since Debreu (1954) that no utility function can represent lexicographic preferences over two or more real-valued attributes. However, it is possible to obtain a representation using a two-function order homomorphism; that is, a pair of real-valued functions  $u$  and  $v$  satisfying  $u(x) > v(y)$  if and only if  $x$  is lexicographically preferred to  $y$  (Fishburn 1974, Bridges 1983, Chateauneuf 1987). Wakker (1988) obtained the necessary and sufficient conditions for the existence of an order homomorphism from a given binary relation to the lexicographic order on  $\mathbb{R} \times \{0, 1\}$ . Knoblauch (2000) described preference representation via order homomorphisms to the lexicographic order on Euclidean  $n$ -space. Martignon and Hoffrage (1999, 2002) described how a linear compensatory model can represent lexicographic preferences over discrete or discretized attributes with finite numbers of levels. Kohli and Jedidi (2007) obtained the necessary and sufficient conditions under which a linear utility function over discrete attributes represents lexicographic preferences. They showed that a number system in which the radix changes from one digit to another is sufficient for representing lexicographic preferences. Tversky (1972) introduced elimination by aspects, a probabilistic lexicographic rule that allows violations



of order independence (and its special case, independence of irrelevant alternatives).

## 2.2. Linear Ordering Problem

We show that the lexicographic inference problem generalizes the linear ordering problem. The latter problem combines multiple rankings or paired comparisons of alternatives into a single representative ordering. It has been studied in economics, psychology, statistics, operations research, and computer science. Well-known applications include the ranking of political candidates by combining voter preferences and the ranking of players/teams in a sport based on the results of matches. Kemeny (1959) described an optimization version of the linear ordering problem. Its objective is to find an ordering of alternatives that is consistent with the largest possible number of preference comparisons between pairs of alternatives (consistent pairs are called nonreversals, and inconsistent pairs are called reversals). Bartholdi et al. (1989) showed that the Kemeny problem is NP-hard. We show that it corresponds to a special case of the lexicographic inference problem.

Martí and Reinelt (2011) and Charon and Hurdy (2007, 2010) reviewed a number of methods for solving the linear ordering problem. These include cutting plane methods (Grötschel et al. 1984), tabu search (Laguna et al. 1999), memetic algorithms (Schiavinotto and Stützle 2004), variable neighborhood search (Garcia et al. 2006), simulated annealing (Charon and Hudry 2007), scatter search and greedy randomized adaptive search (Campos et al. 2001). Ailon et al. (2008) obtained a randomized algorithm that finds a linear ordering for which the expected number of reversals in a minimization version of the problem is no greater than  $11/7$  times the number of reversals in an optimal ordering. Van Zuylen and Williamson (2007) described a polynomial time-approximation algorithm for which the number of reversals is no more than  $8/5$  times the number of reversals in an optimal ordering.

## 2.3. Randomized Algorithms

Motwani and Raghavan (1995) describe the basic concepts in the design and analysis of randomized algorithms. They observe that for many problems a randomized algorithm is the simplest algorithm available or the fastest or both. As noted in the introduction, there are two types of randomized algorithms. A Las Vegas algorithm always finds an optimal solution, but its running time can differ from one run to another. A Monte Carlo algorithm does not always find the optimal solution but has a known running time. We use a Monte Carlo algorithm in this paper.

The probability with which a randomized algorithm selects a solution is often based on the solution to a related optimization problem that can be

efficiently solved. For example, a 0–1 linear integer program may be computationally difficult, but relaxing the integer constraints on the decision variables results in a linear program that can be solved in polynomial time. The fractional solution values may then be used as probabilities when assigning zero or one values to the decision variables. Following Goemans and Williamson (1995), semidefinite programming relaxations have also been used to design randomized algorithms.

We use a different method for generating randomized solutions to the lexicographic inference problem. Let  $n$  denote the total number of levels across all attributes in a problem. We assign a distinct (index) number  $j = 1, \dots, n$  to each level. Let  $u_j = v_j + \epsilon_j$  denote the utility of the level (that is assigned the index)  $j$ , where  $v_j$  is a deterministic component and  $\epsilon_j$  is a stochastic component. We assume that each  $\epsilon_j$  has an independent extreme value distribution. We maximize a likelihood function to estimate the  $v_j$  values from paired comparisons data. This problem can be solved in polynomial time. Given the  $v_j$  values, we implement a single run of the Monte Carlo algorithm by (1) generating values  $u_j = v_j + \epsilon_j$  for all  $j = 1, \dots, n$ ; (2) arranging the  $u_j$  values in a decreasing order  $u_{j_1} > \dots > u_{j_n}$ ; and (3) using the sequence of attribute levels  $j_1, \dots, j_n$  in a lexicographic rule to evaluate alternatives. Each Monte Carlo run can obtain a different solution because it uses different  $\epsilon_j$  values.

We also consider the problem of maximizing the expected value of the solution obtained by a randomized algorithm. We show that this problem is equivalent to the lexicographic inference problem. Thus, by considering a randomized algorithm, we reformulate a constrained, discrete optimization problem as an unconstrained, continuous optimization problem. This allows us to use nonlinear optimization methods to find at least locally optimal solutions to the problem.

We obtain a relation between the maximum likelihood solution and the lower bound on the performance ratio of the associated randomized algorithm. This relation between methods of statistical inference and discrete optimization appears to be new to the literature. In statistics, the objective is to make inferences about population parameters from sample data. There is no estimation problem in the present context. Instead, the objective is to use the data for a problem instance to parameterize a randomized algorithm. The proposed approach is useful if, as in the present problem, the likelihood function can be maximized in polynomial time. In Section 6, we discuss how the present approach can be extended to some other combinatorial optimization problems.

### 3. Formulation and a Randomized Algorithm

We begin by formulating the lexicographic inference problem as a discrete optimization problem. We show that it generalizes the NP-hard linear ordering problem. Then we describe a randomized algorithm and show that maximizing the expected value of its solution is equivalent to solving the lexicographic inference problem.

#### 3.1. Discrete Formulation

As noted in Section 2.1, a lexicographic rule uses a sequence of attribute levels to evaluate alternatives: one alternative is preferred to another if it is better on the most important attribute level on which the two alternatives are different. The data for the lexicographic inference problem are preferences over pairs of alternatives, which may be directly obtained from consumers or inferred from their choices or rankings of alternatives (Kohli and Jedidi 2007, Yee et al. 2007). The objective of the problem is to find an ordering of the attribute levels for which a lexicographic rule correctly predicts the preferences for a maximum number of paired comparisons.

Consider a set of alternatives, each described using  $t \geq 2$  discrete attributes. Let attribute  $l$  have  $n_l \geq 2$  levels for all  $l = 1, \dots, t$ . Some attributes, such as price, may have ordered levels so that a lower price is preferred to a higher price. Other attributes, such as brand name, may be nominal, in which case each level indicates a different brand. Let  $n = n_1 + \dots + n_t$  denote the number of levels across the attributes. We assign a single index  $j$  to each of the  $n$  levels across the attributes. For example, suppose attribute 1 has  $n_1 = 2$  levels and attribute 2 has  $n_2 = 3$  levels. We assign the indices  $j = 1, 2$  to the two levels of attribute 1 and  $j = 3, 4, 5$  to the three levels of attribute 2. In general, we assign the indices  $n_{l-1} + 1, \dots, n_{l-1} + n_l$  to the  $n_l$  levels of attribute  $l$ , where  $n_0 = 0$  and  $l = 1, \dots, t$ .

Let  $j_k$  denote the index of the  $k$ th most important attribute level in a lexicographic rule. We call the ordered vector  $s = (j_1, \dots, j_n)$  a (lexicographic) *sequence*. Strictly speaking, each element of  $s$  is an index associated with an attribute level; for brevity, we refer to it as an attribute level. For example, suppose there are  $n = 4$  attribute levels. Then the sequence  $s = (4, 3, 2, 1)$  characterizes a lexicographic rule that uses  $j_1 = 4$  as the most important attribute level,  $j_2 = 3$  as the second most important attribute level,  $j_3 = 2$  as the third most important attribute level, and  $j_4 = 1$  as the least important attribute level.

Let  $S$  denote the set of all  $n!$  possible sequences of the  $n$  attribute levels. Let  $r = (i, h)$  denote that alternative  $i$  is preferred to alternative  $h$  in a paired comparison.

Let  $R$  denote the set of paired comparisons and  $N = |R|$  the total number of paired comparisons. We formulate the lexicographic inference problem as a 0–1 integer program in which the objective is to select a sequence  $s \in S$  for which a lexicographic rule correctly predicts the preferences for a maximum number of the pairs in  $R$ .

Let  $x_{jk} = 1$  if level  $j$  is the  $k$ th most important attribute level (that is, if it is the  $k$ th element in sequence  $s$ ); otherwise,  $x_{jk} = 0$ . Then

$$\sum_{k=1}^n x_{jk} = 1, \text{ for all } 1 \leq j \leq n,$$

because each attribute level can be assigned exactly one position in a sequence  $s$ . Similarly,

$$\sum_{j=1}^n x_{jk} = 1, \text{ for all } 1 \leq k \leq n,$$

because each position in a sequence  $s$  can be assigned to exactly only one attribute level.

Let  $a_{ij} = 1$  if attribute level  $j$  appears in alternative  $i$  and  $a_{ij} = 0$  otherwise for all  $j = 1, \dots, n$ . The vector  $(a_{i1}, \dots, a_{in})$  is called the *profile* of alternative  $i$ . It has  $t$  nonzero elements, each corresponding to an attribute level that appears in alternative  $i$ . Let

$$b_{ik} = \sum_{j=1}^n a_{ij} x_{jk}, \quad 1 \leq i \leq m, \quad 1 \leq k \leq n.$$

Then  $b_{ik} = a_{ij}$  when level  $j$  is the  $k$ th element in sequence  $s \in S$ ; that is, where attribute level  $j$  is the  $k$ th most important level in a lexicographic rule. The vector  $(b_{i1}, \dots, b_{in})$  rearranges the elements of  $(a_{i1}, \dots, a_{in})$  in order of decreasing importance.

Consider a paired comparison  $(i, h) \in R$ . We say that a lexicographic rule obtains a (preference) reversal if it predicts that  $h$  is preferred to  $i$ ; otherwise, it obtains a nonreversal.

Let

$$b_i = \sum_{k=1}^n \frac{b_{ik}}{2^k}, \quad 1 \leq i \leq m.$$

That is,  $b_i$  is an  $n$ -digit binary number. A number system arranges digits in lexicographic order. Following Kohli and Jedidi (2007), a lexicographic rule that uses attribute level  $j$  as the  $k$ th most important level obtains a nonreversal only if

$$b_i - b_h = \sum_{k=1}^n \sum_{j=1}^n \frac{1}{2^k} (a_{ij} - a_{hj}) x_{kj} \geq 0.$$

Observe that  $b_i - b_h$  is also a binary number. It is nonnegative only if  $(a_{ij} - a_{hj}) x_{kj}$  is nonnegative in its most significant digit. We use this property to formulate the

lexicographic inference problem as the following 0–1 integer programming problem,  $P_1$ .

$$(P_1) \quad \text{Maximize } z = \sum_{r \in R} z_r$$

$$\text{subject to: } 1 + \sum_{k=1}^n \sum_{j=1}^n \frac{1}{2^k} (a_{ij} - a_{hj}) x_{jk} \geq z_{ih},$$

$$\text{for all } (i, h) \in R$$

$$\sum_{k=1}^n x_{jk} = 1, \text{ for all } 1 \leq j \leq n$$

$$\sum_{j=1}^n x_{jk} = 1, \text{ for all } 1 \leq k \leq n$$

$$z_{ih} \in \{0, 1\}, \text{ for all } r = (i, h) \in R$$

$$x_{jk} \in \{0, 1\}, \text{ for all } 1 \leq j, k \leq n.$$

The first constraint allows  $z_{ih} = 1$  only if  $b_i \geq b_h$ ; otherwise,  $z_{ih} = 0$ . The second and third constraints ensure that each attribute level is assigned one position in the importance ordering. The objective function maximizes the number of nonreversals across all paired comparisons. Let  $z^*$  denote the value of the optimal solution to problem  $P_1$ .

### 3.2. Computational Complexity

Schmitt and Martignon (2006) considered a generalization of the lexicographic inference problem in which there are no constraints on the number of attribute levels that are used to describe an alternative. They showed that the problem is NP-hard and that, unless  $P=NP$ , it can have no constant-factor approximation algorithm that runs in polynomial time. We show that the lexicographic inference problem in which each alternative has one level per attribute is also NP-hard but that it allows polynomial time-approximation algorithms with nontrivial performance bounds. (Proofs of lemmas and theorems appear in Appendix A.)

**Theorem 1.** *The lexicographic inference problem is NP-hard.*

The proof of Theorem 1 relies on showing that the Kemeny problem, described in Section 2.2, can be transformed into the lexicographic inference problem in a polynomial number of steps. Because the Kemeny problem is NP-hard, so is problem  $P_1$ . The reduction of the Kemeny problem to the lexicographic inference problem associates a single, unique attribute level with each alternative. As a result, the lexicographic inference problem is NP-hard regardless of the types of (ordered or nominal) attributes and the number of levels per attribute.

### 3.3. Continuous Formulation

Consider a randomized (Monte Carlo) algorithm that selects sequence  $s \in S$  with probability  $p(s)$ , where

$\sum_{s \in S} p(s) = 1$ . For a paired comparison  $r = (i, h)$ , let  $z_r(s) = 1$  if a lexicographic rule using sequence  $s$  correctly predicts that alternative  $i$  is preferred to alternative  $h$ ; otherwise,  $z_r(s) = 0$ . Then the number of nonreversals associated with sequence  $s$  is given by

$$z(s) = \sum_{r \in R} z_r(s), \text{ for all } s \in S.$$

The expected value of the solution obtained by the randomized algorithm is

$$E = \sum_{s \in S} p(s) z(s).$$

We associate a random utility  $u_j = v_j + \epsilon_j$  with attribute level  $j$ , where  $v_j$  is a deterministic component and  $\epsilon_j$  a stochastic component with an independent extreme value distribution. Given values of  $v_1, \dots, v_n$ , we implement the following randomized algorithm:

(1) Obtain independent and random draws  $\epsilon_j$ . Calculate  $u_j = v_j + \epsilon_j$  for each  $j = 1, \dots, n$ .

(2) Arrange the  $u_j$  values in a decreasing sequence,  $u_{j_1} > \dots > u_{j_n}$ . Choose sequence  $s = (j_1, \dots, j_n)$  as the solution to the lexicographic inference problem.

The following lemma gives an explicit expression for the probability with which the randomized algorithm chooses a sequence  $s \in S$ .

**Lemma 1** (Beggs et al. 1981). *Let attribute level  $j$  have random utility  $u_j = v_j + \epsilon_j$ , where  $\epsilon_j$  has an independent extreme value distribution for each  $j = 1, \dots, n$ . Then the randomized algorithm selects sequence  $s = (j_1, \dots, j_n) \in S$  with probability*

$$p(s) = p(u_{j_1} > \dots > u_{j_n}) = \prod_{t=1}^{n-1} \frac{e^{v_{j_t}}}{e^{v_{j_t}} + \dots + e^{v_{j_n}}}, \text{ for all } s \in S.$$

Beggs et al. (1981) give a proof of Lemma 1. Kohli and Jedidi (2015) showed that the preceding algorithm describes a probabilistic lexicographic rule equivalent to elimination by aspects (Tversky 1972).

We consider two methods for obtaining the  $v_j$  values. The first maximizes the expected value of the randomized algorithm over the parameters  $v_1, \dots, v_n$ . We consider this method because it yields an exact continuous formulation of the lexicographic inference problem. The second method, discussed in Section 4, uses maximum likelihood to estimate these parameter values. Its advantage is that the likelihood function can be maximized in polynomial time, thus allowing for a practical approximation algorithm for large problems.

Consider the following problem,  $P_2$ , which maximizes the expected value of the solution obtained by the preceding randomized algorithm.

$$(P_2) \quad \text{Maximize } E = \sum_{s \in S} p(s) z(s).$$

We prove the following result.

**Theorem 2.** Sequence  $s^* \in S$  is an optimal solution to problem  $P_1$  if and only if it is also an optimal solution to problem  $P_2$ .

Formulating the lexicographic inference problem as problem  $P_2$  is useful for proving that it is NP-hard. However, this formulation is not useful for solving the problem because it expresses the expected value as a sum over all  $n!$  sequences in  $S$ . We obtain an alternative expression for the expected value that does not require the explicit enumeration of all the sequences.

Let  $L = \{1, \dots, n\}$  denote the set of  $n$  attribute levels. For each paired comparison  $r = (i, h) \in R$ , let  $L_1^r$  denote the subset of  $m_{1r} = |L_1^r|$  attribute levels with the values one in  $i$  and zero in  $h$ . Similarly, let  $L_2^r$  denote the subset of  $m_{2r} = |L_2^r|$  attribute levels with the values zero in  $i$  and one in  $h$ . Let  $L_{12}^r = L_1^r \cup L_2^r$ . Then  $L_{12}^r$  has  $m_{1r} + m_{2r}$  elements. Let  $L_0^r = L \setminus L_{12}^r$  denote the subset of attribute levels that are both one or both zero in alternatives  $i$  and  $h$ . We make the following observations.

(1)  $m_{1r} = m_{2r}$  because each alternative has one level of each attribute. To see this, suppose the alternatives  $i$  and  $h$  in a pair  $r = (i, h)$  differ on  $t' \leq t$  attributes. Then alternative  $i$  has  $m_{1r} = t'$  attribute levels that do not appear in alternative  $h$  (that is,  $a_{ij} = 1$  and  $a_{hj} = 0$  for these levels), and alternative  $h$  has  $m_{2r} = t'$  attribute levels that do not appear in alternative  $i$  (that is,  $a_{ij} = 0$  and  $a_{hj} = 1$  for these levels).

(2) Regardless of its position in a sequence  $s \in S$ , every attribute level  $j \in L_0^r$  has no effect on the reversal or nonreversal of the paired comparison  $r$ .

(3) If a sequence  $s \in S$  has an attribute level  $j \in L_1^r$  that appears before all attribute levels in  $L_2^r$ , then a nonreversal is obtained for the paired comparison  $r$ .

(4) If a sequence  $s \in S$  has an attribute level  $j \in L_2^r$  that appears before all attribute levels in  $L_1^r$ , then a reversal is obtained for the paired comparison  $r$ .

Because each  $\epsilon_j$  has an extreme value distribution, attribute level  $j \in L_1^r$  precedes all other attribute levels  $\ell \in L_1^r \cup L_2^r$ ,  $j \neq \ell$ , with the multinomial logit probability

$$p(u_j > u_\ell, \text{ for all } \ell \in L_{12}^r) = \frac{e^{v_j}}{\sum_{\ell \in L_{12}^r} e^{v_\ell}}.$$

Thus, the probability of obtaining a nonreversal for the paired comparison  $r \in R$  is

$$\begin{aligned} p(r) &= \sum_{j \in L_1^r} p(u_j > u_\ell, \text{ for all } \ell \in L_{12}^r) = \sum_{j \in L_1^r} \frac{e^{v_j}}{\sum_{\ell \in L_{12}^r} e^{v_\ell}} \\ &= \frac{\sum_{j \in L_1^r} e^{v_j}}{\sum_{\ell \in L_{12}^r} e^{v_\ell}}, \text{ for all } r \in R \end{aligned}$$

and the expected value of the solution obtained by the randomized algorithm is

$$\begin{aligned} E &= \sum_{r \in R} [p(r) \cdot 1 + (1 - p(r)) \cdot 0] = \sum_{r \in R} p(r) \\ &= \sum_{r \in R} \sum_{j \in L_1^r} \frac{e^{v_j}}{\sum_{\ell \in L_{12}^r} e^{v_\ell}}. \end{aligned}$$

This expression for  $E$  does not require enumerating all the sequences in  $S$ . Problem  $P_2$  can be restated as the following nonlinear optimization problem  $P_3$  in which the decision variables are the parameters  $v_1, \dots, v_n$ .

$$(P_3) \quad \text{Maximize } E = \sum_{r \in R} \sum_{j \in L_1^r} \frac{e^{v_j}}{\sum_{\ell \in L_{12}^r} e^{v_\ell}}.$$

Problems  $P_1$  and  $P_3$  are very different formulations of the lexicographic inference problem. Problem  $P_1$  has a linear objective function,  $O(m^2 + n^2)$  0–1 decision variables and  $O(m^2 + n)$  linear constraints, where  $m$  is the number of alternatives and  $n$  is the total number of attribute levels. Problem  $P_3$  has a nonlinear objective function,  $n$  continuous decision variables, and no constraints. Because the lexicographic inference problem is NP-hard, problem  $P_3$  cannot be solved in polynomial time unless  $P=NP$ . However, we can use continuous (Newton or quasi-Newton) methods to obtain at least locally optimal solutions to the problem.

#### 4. Randomized Algorithm Using Maximum Likelihood

Another method for implementing the randomized algorithm is to use the values of  $v_1, \dots, v_n$  obtained by maximizing the likelihood function

$$\mathcal{L} = \prod_{r \in R} p(r).$$

This is equivalent to maximizing the log-likelihood function

$$\ln \mathcal{L} = \sum_{r \in R} \ln p(r),$$

where

$$\ln p(r) = \ln \left( \sum_{j \in L_1^r} e^{v_j} \right) - \ln \left( \sum_{\ell \in L_{12}^r} e^{v_\ell} \right).$$

Maximizing  $\ln \mathcal{L}$  is a convex optimization problem and can be solved in polynomial time.

Note that in statistical inference, the purpose of maximizing a likelihood function is to estimate the population parameters from a sample of observations. Our purpose is different. We wish to obtain the values of  $v_j$  so that we can implement the randomized algorithm for



a particular problem instance. The best solution across multiple runs of the algorithm can be retained as the solution for a problem instance. This solution can also be used to initialize a nonlinear programming procedure for solving problem  $P_3$ . Algorithm 1 gives an overview of the proposed method used for inferring lexicographic rules.

**Algorithm 1** (Procedure for Lexicographic Inference)

1. Maximize likelihood function to estimate  $\hat{v}_1, \dots, \hat{v}_n$
2. **For** iteration  $\iota = 1, \dots, I$ , **do** ▷ randomized algorithm
3.   **For** attribute level  $j = 1, \dots, n$ , **do**
4.     Sample  $\epsilon_{ij} \sim \text{extreme value } (0,1)$
5.     Calculate  $\hat{u}_{ij} = \hat{v}_j + \epsilon_{ij}$
6.   **End For**
7.   If  $u_{j_{i1}} > \dots > u_{j_{in}}$ , set  $s_i = (j_{i1}, \dots, j_{in})$
8.   Calculate  $z(s_i)$ , the number of nonreversals for sequence  $s_i$
9. **End For**
10. Set  $s_{\max} = \arg \max\{z(s_i), \iota = 1, \dots, I\}$  ▷ best randomized solution
11. Solve problem  $P_3$  using the sequence  $s_{\max}$  as a starting solution

Let  $v_j = \hat{v}_j, j = 1, \dots, n$  denote the  $v_j$  values obtained by maximizing the likelihood function. Let  $\mathcal{L}^*$  denote the maximum likelihood value. Then  $\mathcal{L}^*$  is related as follows to the expected value of the solution obtained by a randomized algorithm that uses the parameters  $v_j = \hat{v}_j$ . Let

$$a = \frac{E}{N} = \frac{1}{N} \sum_{r \in R} p(r)$$

denote the arithmetic mean and

$$g = \prod_{r \in R} p(r)^{1/N}$$

denote the geometric mean of the nonreversal probabilities  $p(r)$  across all paired comparisons. Then  $a \geq g$  because an arithmetic mean is no smaller than a geometric mean of a set of nonnegative numbers. Let

$$a^* = \frac{z^*}{N}, \quad \hat{a} = \frac{1}{N} \sum_{r \in R} \hat{p}(r) \quad \text{and} \quad \hat{g} = (\mathcal{L}^*)^{1/N} = \prod_{r \in R} \hat{p}(r)^{1/N},$$

where  $z^*$  is the value of the optimal solution to the lexicographic inference problem and  $N$  is the total number of paired comparisons. Let

$$\hat{p}(r) = \sum_{j \in L_1^r} \frac{e^{\hat{v}_j}}{\sum_{\ell \in L_1^r} e^{\hat{v}_\ell}}$$

denote the maximum likelihood probability of obtaining a nonreversal for the paired comparison  $r \in R$ . Then

$$z^* = Na^* \geq N\hat{a} \geq N\hat{g} = N(\mathcal{L}^*)^{1/N}.$$

We use this sequence of inequalities, together with the following three lemmas, to characterize the performance of a randomized algorithm that uses the maximum likelihood values  $\hat{v}_1, \dots, \hat{v}_n$ .

**Lemma 2.** *The optimal solution to the lexicographic inference problem with  $N$  paired comparisons is no smaller than  $N/2$ .*

**Lemma 3.** *The geometric mean of the maximum likelihood probabilities has the lower bound  $\hat{g} = (\mathcal{L}^*)^{1/N} \geq 1/2$ .*

**Lemma 4** (Pinelis 2015).

$$\hat{a} \geq \hat{g} + \min \left\{ 2\sigma^2, \frac{\kappa^2 \sigma^2}{\kappa^2 - \sigma^2} \right\},$$

where

$$\sigma^2 = \frac{1}{N} \sum_{i \in R} \left( \sqrt{\hat{p}(r)} - \mu \right)^2, \quad \kappa^2 = \frac{1}{N} \sum_{r \in R} \left( \sqrt{\hat{p}_{\max}} - \sqrt{\hat{p}(r)} \right)^2,$$

$$\mu = \frac{1}{N} \sum_{r \in R} \sqrt{\hat{p}_i} \quad \text{and} \quad \hat{p}_{\max} = \max_{i \in R} \hat{p}(r).$$

Pinelis (2015) gives a proof of Lemma 4 in which the  $\hat{p}(r)$  values may be any nonnegative numbers, not just probabilities.

Theorem 3 uses Lemmas 2–4 to obtain a lower bound on the performance ratio of a randomized algorithm that uses the maximum likelihood values  $\hat{v}_1, \dots, \hat{v}_n$ .

**Theorem 3.** *Let  $\phi = N\hat{a}/z^*$  denote the performance ratio of a randomized algorithm using the maximum likelihood solution  $v_j = \hat{v}_j, j = 1, \dots, n$ . Then*

$$\phi \geq K^* \left[ \hat{g} + \min \left\{ 2\sigma^2, \frac{\kappa^2 \sigma^2}{\kappa^2 - \sigma^2} \right\} \right],$$

where  $\hat{g} \geq 1/2$  and  $K^* = N/z^* \geq 1$ .

In Theorem 3,  $\hat{g} \geq 1/2$  follows from Lemma 3 and  $K^* \geq 1$  because  $z^* \leq N$ . The value of  $z^*$  is not known but can be replaced by an upper bound in the expression for  $K^*$ . A trivial upper bound is  $z^* = N$ . A possible improvement in the upper bound is obtained as follows. Let  $n_{ih}$  denote the number of paired comparisons in which alternative  $i$  is preferred to alternative  $h$ . Then any feasible solution, including the optimal solution, can obtain at most  $\max\{n_{ih}, n_{hi}\}$  nonreversals for the pair of alternatives  $i$  and  $h$ . Thus,

$$z^* \leq M = \sum_{(i,h)} \max\{n_{ih}, n_{hi}\} \leq N.$$

Another upper bound for  $z^*$  can be obtained by solving a linear programming relaxation of problem  $P_1$ . Any such upper bound for  $z^*$  can be used to obtain an instance-specific lower bound for  $\phi$ .



As with the greedy algorithm considered by Yee et al. (2007) and Kohli and Jedidi (2007), the maximum likelihood solution obtains the optimal solution to the lexicographic inference problem when all paired comparisons can be perfectly predicted by a lexicographic rule. In this case,  $\hat{g} = \hat{a} = 1$  because  $p(r) = 1$  for all  $N$  paired comparisons, and so  $\mathcal{L}^* = 1$ . It follows that  $K^* = N/z^* = 1$  and  $\phi = K^*\hat{g} = K^*\mathcal{L}^{*1/N} = 1$ . The maximum likelihood solution also obtains the optimal solution when  $z^* = N/2$ . In this case,  $K^* = N/z^* = 2$ . Because  $g^* \geq 1/2$ ,  $\phi \geq K^*g^* = 1$ . We summarize these results in the following corollary to Theorem 3.

**Corollary 1.** *The maximum likelihood solution finds an optimal ordering of attribute levels when  $z^* = N/2$  or  $z^* = N$ .*

## 5. Application

We employed the preceding randomized algorithm to identify lexicographic rules used by consumers to choose electronic tablets. A commercial market-research firm provided the data used for the analysis. These data were collected from 137 subjects who were screened to be interested in purchasing an electronic tablet in the next 12 months. Each subject was shown 15 choice sets, which could differ across subjects. All choice sets had three alternatives, none of which dominated the others across attributes. A subject's task was to select one alternative from a choice set. We used the data to obtain aggregate and individual-level parameter estimates using maximum likelihood. The aggregate analysis served as a benchmark for evaluating the individual-level models. It also allowed an assessment of the extent to which aggregate lexicographic models can predict preferences across individuals. We used the maximum likelihood estimates to implement the randomized algorithm  $I = 1,000$  times and selected the solution with the largest number of nonreversals. We used this solution to provide starting values to a nonlinear programming routine for solving problem  $P_3$ .

The tablets shown to participants were selected from 350 product profiles constructed using a fractional factorial plan in which the attributes were used as design factors. A fractional factorial plan has a subset of all possible combinations of design factors. Its key feature is that the subset is chosen to ensure that the parameter estimates (in a linear model) are orthogonal. See Box et al. (1978) for details. The present experimental plan used the following attributes and levels:

- (1) Brand name: iPad, Galaxy, Surface, Nexus, and Kindle
- (2) Screen size: 7, 8, 9, and 10 inches
- (3) Hard-drive capacity: 16, 32, 64, and 128 GB
- (4) RAM: One, two, and four GB
- (5) Battery life: seven, eight, and nine hours
- (6) Price: \$169, \$199, \$299, \$399, and \$499

We used standard dummy variable coding for the logit model. For the randomized algorithm, we treated brand name and screen size as nominal attributes and used "aspect coding" for price, memory, disk size, and battery life. The following example illustrates aspect coding. Let  $x_{i1} = 1$  if the battery life for tablet  $i$  is at least seven hours,  $x_{i2} = 1$  if it is at least eight hours, and  $x_{i3} = 1$  if it is at least nine hours. Then  $x_{i1} = x_{i2} = x_{i3} = 1$  for a tablet with a nine-hour battery;  $x_{i1} = x_{i2} = 1$ ,  $x_{i3} = 0$  for a tablet with an eight-hour battery; and  $x_{i1} = 1$ ,  $x_{i2} = x_{i3} = 0$  for a tablet with a seven-hour battery. Similarly, suppose  $x_{i4} = 1$  if the price of tablet  $i$  is no greater than \$169 and  $x_{i5} = 1$  if it is no greater than \$199. Then  $x_{i4} = x_{i5} = 1$  for a \$169 tablet, and  $x_{i4} = 0$ ,  $x_{i5} = 1$  for a \$199 tablet. Note that for the ordered attributes, the dummy variables representing the least-preferred levels (16-GB hard drive, one-GB RAM, seven-hour battery life, and \$499 price) have values equal to one in every product profile. Consequently, the parameters associated with these levels are not identified. However, the parameter associated with each level of a nominal attribute is identified.

### 5.1. Estimation and Validation

We generated paired comparisons between the chosen alternative and the other two alternatives for each of the  $15 \times 137 = 2,055$  choice sets.<sup>1</sup> We used the  $2 \times 2,055 = 4,110$  paired comparisons to estimate the various models. We randomly selected two paired comparisons per individual for model validation. We reestimated the models using the remaining  $2 \times 14 \times 137 = 3,836$  paired comparisons and used the results to make out-of-sample predictions for the  $2 \times 137 = 274$  holdout paired comparisons. We repeated the procedure 100 times for each aggregate model described herein. We also repeated the procedure 100 times for the mixed logit model. For the other individual-level methods, we held out two randomly selected paired comparisons per person and replicated the procedure 10 times; this yielded  $2 \times 10 \times 137 = 27,400$  paired comparisons for validation across replications and subjects. In each case, we report the average out-of-sample hit rates across replications.

We used the following five methods for inferring lexicographic rules. The first two are proposed methods, and the next three are benchmarks. We implemented each method using the R programming language on a Dell laptop computer with an Intel i5-6440HQ CPU (2.60 GHz) and eight GB of RAM.

- (1) The randomized algorithm using the maximum likelihood values  $\hat{v}_1, \dots, \hat{v}_n$ . We obtained  $n$  independent random draws  $\epsilon_1, \dots, \epsilon_n$  from an extreme value distribution and calculated  $\hat{u}_j = \hat{v}_j + \epsilon_j$  for each  $j = 1, \dots, n$ . If  $\hat{u}_{j_1} \geq \dots \geq \hat{u}_{j_n}$ , we used the lexicographic sequence  $s = (j_1, \dots, j_n)$  as the randomized solution. We repeated

the procedure 1,000 times and selected a solution for which the number of nonreversals,  $z(s)$ , was the highest.

(2) The solution obtained by maximizing the expected value of a randomized algorithm. We used the best maximum likelihood solution as the starting solution. We used the limited memory version of the Broyden–Fletcher–Goldfarb–Shanno algorithm, available in the RStan package in R, to maximize the expected value (Byrd et al. 1994). The algorithm uses a quasi-Newton method and is suited to problems with large numbers of decision variables.

(3) A probabilistic lexicographic rule, the solution to which is the ordering of attribute levels obtained by arranging the maximum likelihood values  $\hat{v}_1, \dots, \hat{v}_n$  in a decreasing sequence.

(4) The greedy algorithm described by Yee et al. (2007) and Kohli and Jedidi (2007).

(5) A local search algorithm that begins with an arbitrary sequence  $s$ , exchanges the positions of a pair of attribute levels if this improves the solution value, and repeats this exchange until no further improvement in the solution value can be obtained (see Papadimitriou and Steiglitz (1998) for a discussion of local search). We implemented the local search algorithm 100 times, each time using a random initial sequence. We report the best solution value across the 100 runs.

(6) A logit model estimated using paired comparisons for consistency with the other methods (virtually identical results were obtained using the choice data). We used maximum likelihood to estimate the aggregate-level parameters and a hierarchical Bayesian approach to estimate the individual-level parameters. (Individual-level estimates using only the 15 choices—30 paired comparisons—per person overfit the data, obtaining almost perfect in-sample hit rates but an average out-of-sample hit rate of only 0.76 for the logit model.) We implemented the hierarchical Bayesian approach by assuming that the prior distribution of the population means was normal with mean zero and variance 100 (a large value) and that the prior of the covariance matrix had an inverse Wishart distribution. We used Hamiltonian Monte Carlo (HMC) to obtain sample draws for the parameter estimates. HMC is similar to the Metropolis–Hasting algorithm but uses Hamiltonian dynamics for sampling and converges more quickly to the mode of the joint posterior distribution. Appendix B gives a brief overview of HMC.

## 5.2. Results

Table 1 reports the parameter estimates for the aggregate models. It also reports the importance weights for the attribute levels in the randomized algorithm. For the nominal attributes, brand name and screen size, the importance weight of a level is defined to be equal to the exponential of its parameter estimate. For the

ordinal attributes, the importance weight of a level reflects the aspect coding. For example, consider hard disk size. Its base level of 16 GB has zero importance weight. The next highest level (32 GB) has a parameter estimate of 1.39 in the aggregate model and, thus, an importance weight of  $e^{1.39} = 4.01$ . The 64-GB hard drive satisfies the two conditions “no less than 32 GB” and “no less than 64 GB.” Its parameter estimate in the aggregate model is 1.79, and thus, its importance weight is  $e^{1.39} + e^{1.79} = 10.00$ . The results suggest a greater marginal change in the importance weight at higher prices. For example, a price decrease from \$399 to \$299 increases the importance weight from  $e^{2.13} = 8.41$  to  $e^{2.13} + e^{2.50} = 20.60$ .

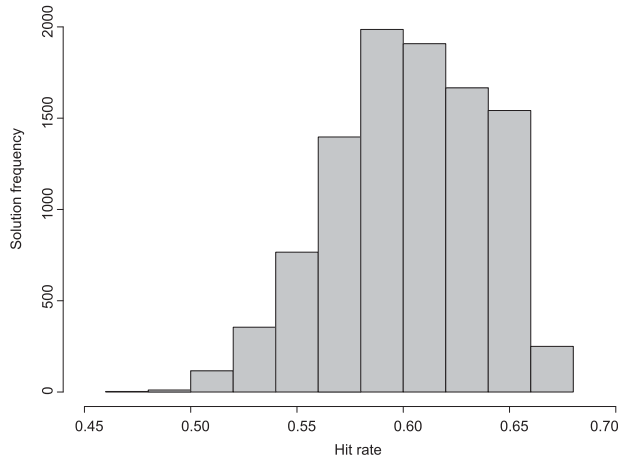
**5.2.1. Aggregate Solutions.** All logit parameter estimates, except for “Surface,” are statistically significant at the 5% confidence level. The values of the estimates suggest that increasing battery life from eight hours to nine hours and hard drive from 64 GB to 128 GB provides no incremental value to a consumer; that a nine-inch screen is preferred to a 10-inch screen; and

**Table 1.** Aggregate-Level Parameter Estimates for the Logit Model and the Randomized Algorithm

Attribute	Logit estimate	Randomized algorithm estimate	Importance weight
Brand			
Nexus	0	0	1.00
Kindle	0.24	0.78 <sup>a</sup>	2.18
iPad	1.01	2.88	17.81
Galaxy	0.35	1.55	4.71
Surface	0.14 <sup>a</sup>	0.9 <sup>a</sup>	2.46
Screen size, inches			
7	0	−3.67 <sup>a</sup>	0.03
8	0.22	0.96 <sup>a</sup>	2.61
9	0.46	1.70	5.47
10	0.35	1.39 <sup>a</sup>	4.01
Hard drive, GB			
16	0		
32	0.23	1.39 <sup>a</sup>	4.01
64	0.59	1.79	10.00
128	0.59	−0.16 <sup>a</sup>	10.86
RAM, GB			
1	0		
2	0.31	1.28 <sup>a</sup>	3.60
4	0.65	1.70	9.07
Battery life, hours			
7	0		
8	0.12	0.07 <sup>a</sup>	1.07
9	0.13	−0.01 <sup>a</sup>	2.06
Price, \$			
169	0	1.74	33.68
199	−0.34	2.00	27.99
299	−0.73	2.50	20.60
399	−1.29	2.13	8.41
499	−1.73		

<sup>a</sup>Not significant at the 95% confidence level.

**Figure 2.** Distribution of Hit Rates Across 1,000 Runs of the Randomized Algorithm for Aggregate Data



that iPad is the most-preferred brand, followed by Galaxy, Kindle, and Surface. As expected, the parameter estimates for price become increasingly more negative at higher prices.

Statistical significance of the parameter estimates is not relevant for the randomized algorithm. This is because a randomized algorithm does not have estimates but can be implemented using any set of parameter values. A randomized algorithm using the aggregate parameter values in Table 1 assigns the highest importance weights to the prices (below \$169, below \$199, below \$299), iPad, hard drive (at least 128 GB, then at least 64 GB), and RAM (at least four GB).

Figure 2 shows the distribution of the in-sample hit rates for 1,000 lexicographic rules identified using the randomized algorithm. The worst rule has an in-sample hit rate of 0.48. The best lexicographic rule has the highest in-sample (0.671) and out-of-sample (0.652) hit rates. Table 2 shows the ordering of attribute levels for this solution. The most important attribute levels are the three lowest prices, followed by iPad, 128 GB hard drive, and 64 GB hard drive. The least important attribute levels are eight-hour battery, Nexus, and 7-inch screen. The least-preferred levels of the ordered

**Table 2.** Aggregate-Level Lexicographic Ranking of Attribute Levels Identified by the Randomized Algorithm

Rank	Attribute level	Rank	Attribute level
1	\$169	11	10-inch screen
2	\$199	12	32 GB hard drive
3	\$299	13	2 GB RAM
4	iPad	14	8-inch screen
5	128 GB hard drive	15	Surface
6	64 GB hard drive	16	Kindle
7	4 GB RAM	17	Nine-hour battery
8	\$399	18	Eight-hour battery
9	9-inch screen	19	Nexus
10	Galaxy	20	7-inch screen

**Table 3.** In-Sample and Out-of-Sample Hit Rates for Different Aggregate Models

Aggregate model	In-sample	Out-of-sample
Logit model	0.604	0.602
Greedy algorithm	0.657	0.655
Local search algorithm	0.669	0.668
Probabilistic lexicographic rule	0.652	0.652
Randomized algorithm	0.671	0.652
Maximum expected value solution	0.677	0.676

attributes do not appear in Table 2 because, as noted, their parameters are not identified.

Table 3 compares the hit rates across the different methods used for inferring an aggregate lexicographic rule. The solution obtained by maximizing the expected value has the best in-sample and out-of-sample hit rates. The best solutions obtained by the randomized algorithm and the local search algorithm obtain only slightly lower hit rates. The probabilistic lexicographic rule, which orders attribute levels in decreasing order of the  $\hat{v}_j$  values, performs as well as the greedy algorithm. The logit model had the lowest hit rate.

Table 4 shows the computational time for each of the seven models. The greedy algorithm took the least time of 0.03 seconds. The local search algorithm took 0.1387 seconds per iteration and, thus, a total of 13.87 seconds for the 100 iterations. The maximum likelihood estimation for the probabilistic lexicographic rule took 1.33 seconds. This solution was used to implement the randomized algorithm, which took 0.004 seconds per iteration and 4.14 seconds for the 1,000 iterations. The maximum expected value solution was obtained in 10.58 seconds when it used the best randomized solution as the initial solution. Thus, the total time for obtaining the maximum likelihood parameters, implementing the randomized algorithm and maximizing the expected value was  $1.33 + 4.14 + 10.58 = 16.05$  seconds; 9% of this time was used for obtaining the maximum likelihood parameters, 26% for implementing the randomized algorithm, and 65% for maximizing the expected value.

Recall that  $z^* \leq M = \sum_{(i,h) \in R} \max(n_{ih}, n_{hi})$ . In the present case,  $M = 2,986$ , which implies that the highest possible hit rate is  $2,986/4,110 = 0.73$ . Thus, the in-sample

**Table 4.** Computational Time in Seconds for Different Aggregate Models

Model	CPU time
Logit model	0.18
Greedy algorithm	0.03
Local search algorithm	13.87
Probabilistic lexicographic rule	1.33
Randomized algorithm	4.14
Maximum expected value solution	10.58

**Table 5.** Hit Rates for Different Individual-Level Models

Model	In-sample	Out-of-sample
Mixed logit model	0.844	0.764
Greedy algorithm	0.873	0.784
Local search algorithm	0.915	0.768
Probabilistic lexicographic rule	0.923	0.808
Randomized algorithm	0.947	0.818
Maximum expected value solution	0.949	0.816

hit rate obtained by the randomized algorithm is  $0.68/0.73 = 0.93$  times the maximum achievable hit rate. Because  $M$  is an upper bound for the optimal solution value, the performance ratio of the randomized algorithm is at least 0.93. The Pinelis lower bound for the hit rate is 0.59 ( $\mu = 0.763, \sigma^2 = 0.019, \kappa^2 = 0.067, \hat{g} = 0.559$ ), which is close to the in-sample hit rate obtained by the logit model.

**5.2.2. Individual-level Solutions.** Table 5 reports the in-sample and out-of-sample hit rates for the different individual-level models. We make the following observations:

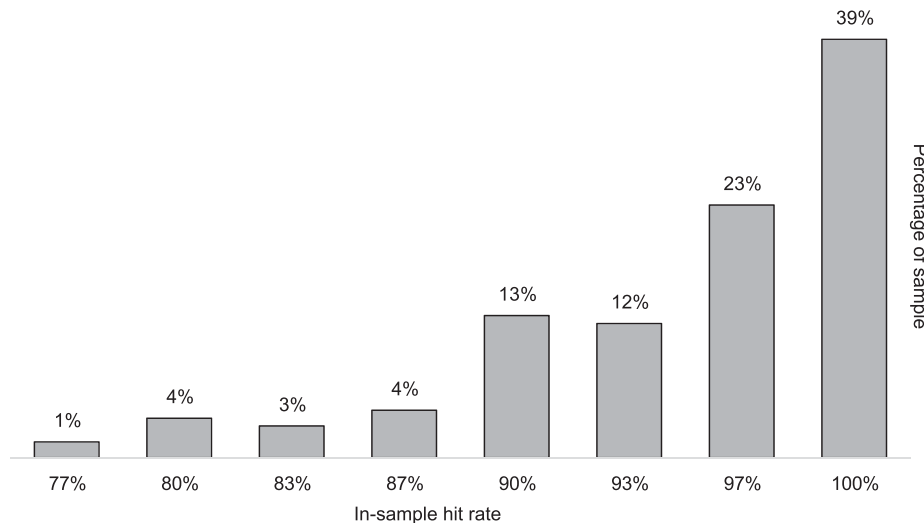
- (1) The mixed logit model has the lowest in-sample and out-of-sample hit rates.
- (2) The greedy algorithm obtains better hit rates than the mixed logit model.
- (3) The local search algorithm obtains a better in-sample hit rate but a slightly lower out-of-sample hit rate than the greedy algorithm.
- (4) The probabilistic lexicographic rule obtains better in-sample and out-of-sample hit rates than the mixed logit model, the greedy algorithm, and local search.
- (5) The randomized algorithm obtains further improvements in the hit rates. The in-sample hit rate (0.95) is slightly higher than the Pinelis lower bound of 0.92

for the randomized algorithm ( $\mu = 0.953, \sigma^2 = 0.013, \kappa^2 = 0.015, \hat{g} = 0.892$ ).

(6) Maximizing the expected value obtains about the same in-sample (0.95) and out-of-sample (0.82) hit rates as the randomized algorithm using maximum likelihood.

Figure 3 shows the distribution of in-sample hit rates for the individual lexicographic rules obtained by the randomized algorithm. These rules correctly predict all paired comparisons for 39% of the sample (53 individuals). The number of incorrectly predicted paired comparisons is one for 23% of the sample (32 individuals), two for 12% of the sample (16 individuals), three for 13% of the sample (18 individuals), and four or more for the remaining 12% of the sample (16 individuals).

Table 6 shows the computational time for each of the six individual-level models. The mixed logit model, estimated across the 137 subjects, had a running time of 1,119.7 seconds (18.67 minutes) for 2,000 HMC draws. The greedy heuristic and the probabilistic lexicographic rule took less than 0.001 second per person, which is less than  $0.001 \times 137 \approx 0.14$  second across the subjects. The running time for the local search algorithm increases linearly with the number of runs. It took 0.0038 second for each run,  $0.0038 \times 100 = 0.38$  second across 100 runs per subject and  $0.38 \times 137 = 52.06$  seconds across the subjects. The randomized algorithm took 0.00051 second per iteration and  $0.00051 \times 1,000 = 0.51$  second across 1,000 iterations. Its running time was  $0.51 \times 137 = 69.87$  seconds (1.165 minutes) across the subjects. Maximizing the expected value took an additional 0.02 second per person and  $0.02 \times 137 = 2.74$  seconds across the subjects. The total time for estimating the maximum likelihood parameters, implementing the randomized algorithm and maximizing the expected value was less than  $0.137 + 69.87 + 2.74 = 72.747$  seconds

**Figure 3.** Distribution of In-Sample Hit Rates for the Individual Lexicographic Rules Obtained by the Randomized Algorithm



**Table 6.** Computational Time in Seconds for Different Individual-Level Models

Model	CPU time
Mixed logit model	1,119.700
Greedy algorithm	≤ 0.14
Local search algorithm	52.06
Probabilistic lexicographic rule	≤ 0.14
Randomized algorithm	69.87
Maximum expected value solution	0.020

(1.21 minutes) across the subjects; 96% of this time was used for the 1,000 runs of the randomized algorithm.

### 5.3. Individual Lexicographic Rules

As noted, the hit rates obtained by the randomized algorithm are practically indistinguishable from those obtained by additionally maximizing the expected value. For this reason, we only examine the solutions obtained by the randomized algorithm.

Figure 4 shows the percentage of individuals using different attribute levels as the first screening criterion. We make the following observations:

(1) The three most commonly used attribute levels were \$399 maximum price (22% of sample), iPad (20% of sample), and \$299 maximum price (19% of sample).

(2) Only 13% of the sample used Kindle, Galaxy, Surface, or Nexus as the first screening criterion.

(3) If tablets with each of the 17 attribute levels were available, their share of purchases would be at least as large as the percentages shown in Figure 4. For example, an iPad would obtain at least 20% share of purchases. Its share could be higher if an individual began with another attribute level at the first stage (say \$169) and then used iPad as a later screening criterion.

The criteria shown in Figure 4 can be grouped into three categories: brand first, price first, and features

first. Figure 5 shows the groups. We make the following observations:

(1) Forty-six percent of the sample used price as the most important lexicographic criterion. Among these consumers, 48% used \$399, 41% used \$299, and 10% used \$199 as the cutoff price. Only 2% would not consider a tablet priced above \$169.

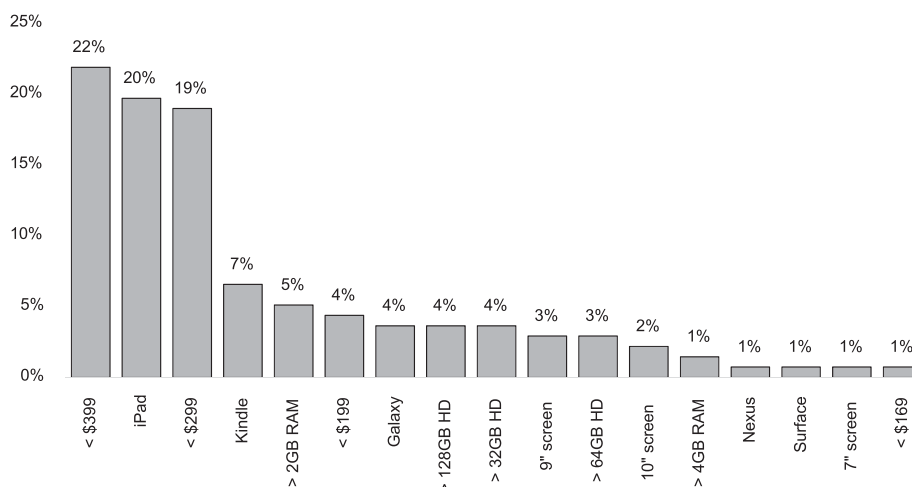
(2) Thirty-one percent of the sample used brand as the first screening criterion. Among these consumers, 63% sought an iPad, 21% Kindle, 12% Galaxy, and 2% Surface and Nexus, each.

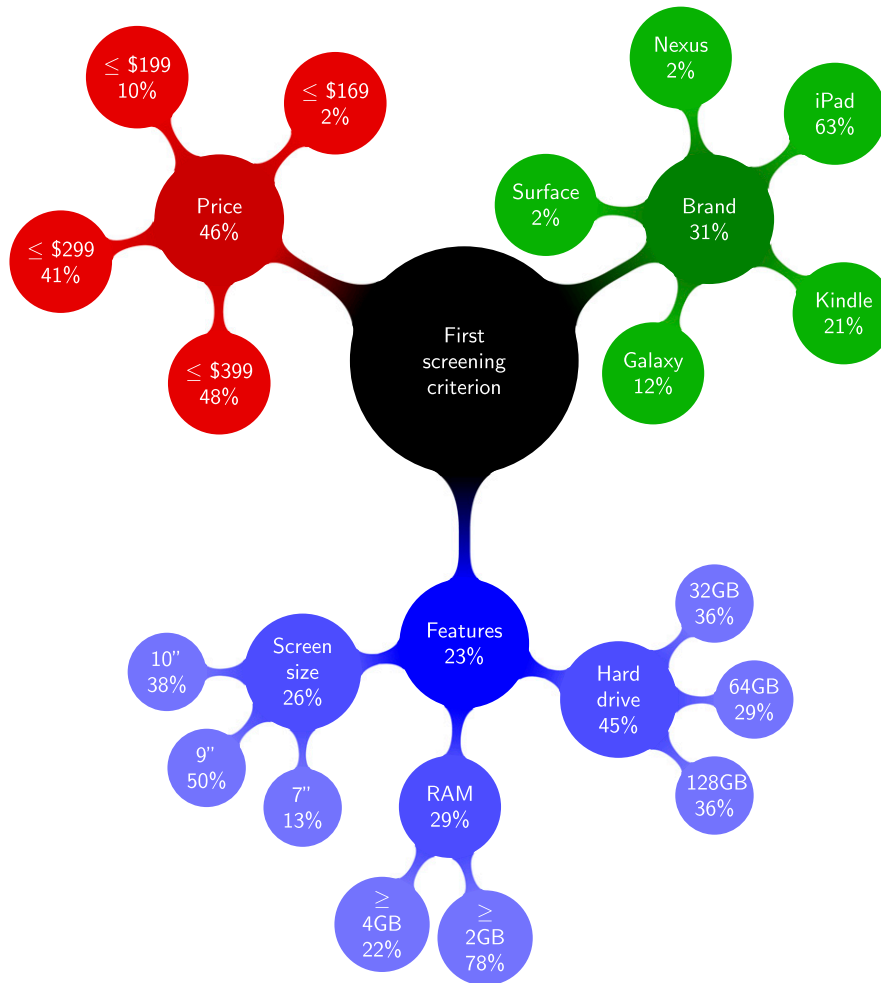
(3) Twenty-three percent of the sample first screened alternatives using a feature other than price or brand. Forty-five percent of these consumers considered hard drive, 29% RAM, and 26% screen size. Among those considering hard drive, 36% sought at least 128 GB; among those considering RAM, 78% sought at least two GB; and among those seeking screen size, 50% sought a 9-inch screen and 38% a 10-inch screen.

Figure 5 can be extended to represent the other criteria used for the sequential screening of alternatives. For example, consider the consumers who used iPad as the first screening criterion. Figure 6 shows the distribution of the criteria they used for a second-stage screening: 33% considered price and screen size each, 19% hard drive, 11% RAM, and 4% battery life. Among those looking at price, 44% would pay at most \$399, 33% at most \$199, and 22% at most \$169. Among the 33% using screen size as the second screening criterion, 44% consider a 9-inch screen, 33% a 10-inch screen, and 22% a 7-inch screen. Similar interpretations can be obtained for the other branches in Figure 6.

We can also examine the effect of an attribute level not being available to consumers. For example, suppose an iPad were not available. Then the 20% consumers who used iPad as the first screening criterion would switch to another feature. Figure 7 shows that

**Figure 4.** Percentage of Individuals Using Different Attribute Levels as the First Screening Criterion in a Lexicographic Rule



**Figure 5.** (Color online) First Criterion Used to Screen Alternatives by Different Individuals

4% of these consumers would switch to at least two GB RAM and at least nine-hour battery life; 15% to hard drive size; 19% to price and screen size; and 41% to another brand. Among the consumers using another brand as the first screening criterion, 73% would look for a Galaxy tablet.

The information in Figures 5 and 7 can be combined to obtain the following observations about how the first-stage screening would change if iPad were not available.

(1) The percentage of consumers using brand as the first screening criterion would decrease from 31% to  $(31 - 20) + (0.41 \times 20) = 19.2\%$ . Galaxy would benefit the most: with iPad available, only 4% of consumers use Galaxy as the first screening criterion; without iPad,  $4 + (20 \times 0.41 \times 0.73) = 9.99\%$  would use Galaxy as the first screening criterion.

(2) The percentage of consumers using price as the first screening criterion would increase from 46% to  $46 + (20 \times 0.19) = 49.8\%$ .

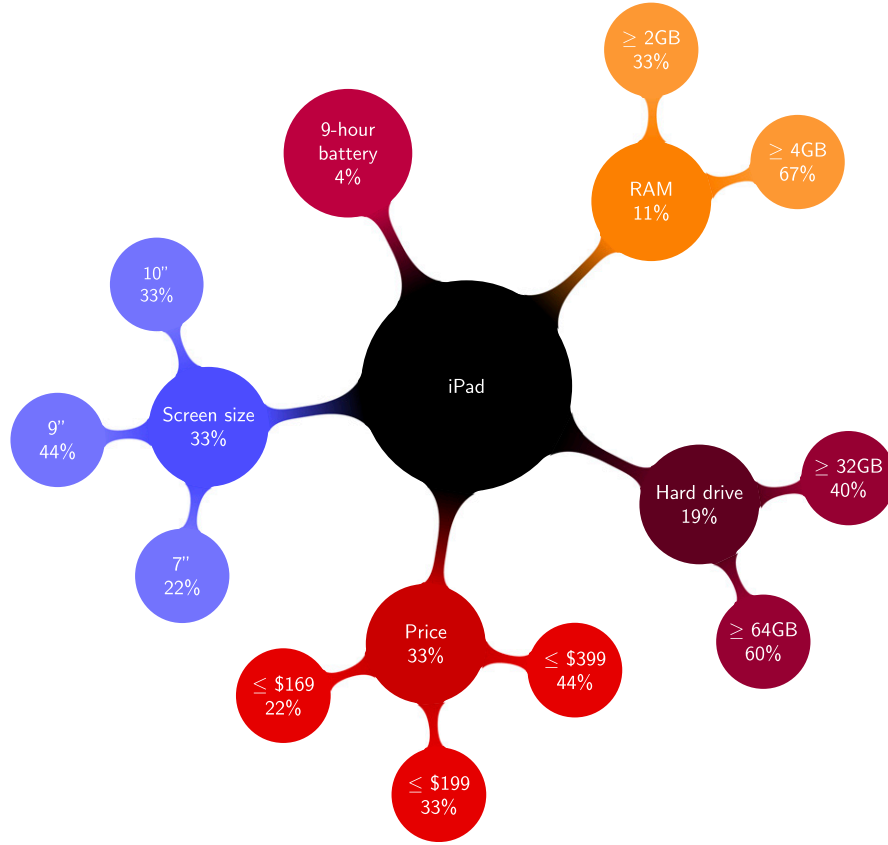
(3) The percentage of consumers using a feature other than brand or price as the first screening criterion would

increase from 23% to  $23 + (20 \times 0.4) = 28\%$ . Hard drive would be used as the first screening criterion by an additional  $20 \times 0.15 = 3\%$  of consumers, screen size by an additional  $20 \times 0.19 = 3.8\%$  of consumers, and each of (at least) nine-hour battery life and (at least) two GB RAM by an additional  $20 \times 0.04 = 0.8\%$  of consumers.

## 6. Conclusion

Inferring a lexicographic rule from paired comparisons, ranking, or choice data is an NP-hard problem. We considered a randomized algorithm and showed that maximizing its expected value is equivalent to solving the lexicographic inference problem. An approximate solution to the problem can be obtained in polynomial time by using a maximum likelihood method to estimate the parameters for the randomized algorithm. The maximum likelihood value is related to a lower bound on the expected value of the solution obtained by the algorithm. We illustrate the proposed approach using data from a choice set experiment for electronic tablets. The solutions obtained in this way are

**Figure 6.** (Color online) Second Criterion Considered by Individuals Who Use iPad as the First Screening Criterion



substantially better than those obtained using a previously proposed greedy algorithm, a local search algorithm, and a logit model.

A useful area of future research is to consider how the design of products and product lines changes when consumers use a lexicographic rule (Bertsimas and Mišić 2019). The present research on the subject mostly assumes that consumers have compensatory preferences (e.g., Green and Krieger 1985, Kohli and Krishnamurti 1987, Chen and Hausman 2000, Belloni et al. 2008, and Bertsimas and Mišić 2017). Another area of research is to develop data-collection methods for inferring lexicographic preferences. Such methods have previously been developed for compensatory preferences by Toubia et al. (2003). DeSarbo et al. (2005) developed dynamic models of utility evolution, examining how the preference structure changes over repeated measurements. Similar methods for lexicographic preferences would be useful.

Methodologically, the absence of constraints in the expected value formulation of the lexicographic inference problem can be useful. As noted, the linear ordering problem, which is a special case of the lexicographic inference problem, has a standard integer programming formulation with  $O(m^2)$  decision variables and  $O(m^3)$  constraints, where  $m$  is the number of

alternatives. Such a problem with 1,000 alternatives has about a million decision variables and a billion constraints. Even representing such a large problem on a computer can be difficult. We have used the nonlinear formulation described in the paper to solve linear ordering problems with up to 2,000 alternatives in minutes on a laptop computer. Similar formulations are feasible for several other discrete optimization problems, including maxcut, minimum vertex cover, and maximum satisfiability problems. For example, given an undirected graph  $G(V, E)$  and nonnegative weights  $w_{ij} = w_{ji}$  on the edges  $(i, j) \in E$ , the maximum cut (maxcut) problem is to find the set of vertices  $S \subset V$  that maximizes the weight of the edges in the cut  $(S, \bar{S})$ , that is, the weight of the edges with one end point in  $S$  and the other in  $\bar{S}$ . Consider a randomized algorithm that assigns vertex  $i$  to subset 1 with probability  $p_i$  and to subset 2 with probability  $1 - p_i$ . Then the probability that an edge between nodes  $i$  and  $j$  appears across the cut is  $p_{ij} = p_i(1 - p_j) + (1 - p_i)p_j$ . Let  $p_i = (1 - y_i)/2$ , where  $-1 \leq y_i \leq 1$ . Then  $p_i = 1$  when  $y_i = -1$ ,  $p_i = 0$  when  $y_i = 1$ , and  $p_{ij} = (1 - y_i y_j)/2$ . Let  $E[z]$  be the expected value of the number of edges across the cut. Then

$$E[z] = \sum_{(i,j) \in E} p_{ij} = \frac{1}{2} \sum_{(i,j) \in E} w_{ij}(1 - y_i y_j).$$

**Figure 7.** (Color online) How the Absence of iPad Affects First-Stage Screening by Individuals Who Would Otherwise Have Looked for an iPad at the First Stage



The problem of maximizing  $E[z]$ , subject to the constraints  $-1 \leq y_i \leq 1$  for all  $i \in V$ , is identical to Goemans and Williamson's formulation for the maxcut problem. The difference is that, in our development,  $y_i$  by definition is linearly related to the probabilities  $p_i$  ( $y_i/2$  is the deviation of  $p_i$  from  $1/2$ ), whereas it is a decision variable with  $\pm 1$  values in Goemans and Williamson's formulation. The corresponding problem of maximizing the likelihood function  $\prod_{(i,j) \in E} w_{ij}(1 - y_i y_j)$  is a convex optimization problem. As in the present paper, the maximum likelihood value can be used to obtain a lower bound for a randomized algorithm.

### Acknowledgments

The authors thank Professor Ramesh Krishnamurti, Simon Fraser University, for showing that the linear ordering problem is a special case of the lexicographic inference problem. They also thank Sawtooth Software for providing the data from the tablet computer study. Finally, they thank the reviewers and the associate editor for their guidance and helpful comments.

### Appendix A. Proofs of Theorems and Lemmas

**Proof of Theorem 1.** We prove the result by showing that Kemeny's formulation of the linear ordering problem can be transformed in a polynomial number of steps into the lexicographic inference problem.

The linear ordering problem is to find a single ranking of a number of political candidates that best represents the preferences of individual voters. Kemeny's version of the problem converts the individual rankings into paired comparisons, where  $r = (i, h)$  if a particular voter prefers candidate  $i$  to candidate  $j$ . Let  $R = \{(i, h)\}$  denote the set of all paired comparisons across candidates and voters. The objective of Kemeny's problem is to find an ordering of the alternatives that maximizes the number of nonreversals across all paired comparisons in  $R$ . Bartholdi et al. (1989) showed that this is an NP-hard problem.

A Kemeny problem with  $m$  alternatives can be represented as the following lexicographic inference problem with  $n = m$  attribute levels. Let  $a_{ij} = 1$  if  $i = j$  and  $a_{ij} = 0$  otherwise for all  $i, j = 1, \dots, m$ . Then

$$b_{ik} = \sum_{j=1}^n a_{ij} x_{jk} = a_{ii} x_{ik},$$

$$b_i = \sum_{k=1}^m \frac{b_{ik}}{2^k} = \sum_{k=1}^m \frac{a_{ii} x_{ik}}{2^k}$$

and

$$b_i - b_h = \sum_{k=1}^m \frac{1}{2^k} (a_{ii} x_{ik} - a_{hh} x_{hk}), \text{ for each } r = (i, h) \in R.$$

Let  $x_{ik} = 1$  for  $k = k_i$  and  $x_{hk} = 1$  for  $k = k_h$ . Then

$$b_i - b_h = \frac{1}{2^{k_i}} - \frac{1}{2^{k_h}}.$$



Thus,  $b_i - b_h > 0$  if  $k_i < k_h$ , and  $b_i - b_h < 0$  otherwise. Equivalently, an ordering of attribute levels obtains a nonreversal of the paired comparison  $(i, h) \in R$  if attribute level  $i$  has a higher rank than attribute level  $h$  in the linear ordering; otherwise, it obtains a reversal. It follows that maximizing the number of nonreversals in the lexicographic inference problem is equivalent to maximizing the number of nonreversals in the Kemeny problem. Because the linear ordering problem is NP-hard, so is the lexicographic inference problem.  $\square$

**Proof of Theorem 2.** Without loss of generality, let  $s^* = (1, \dots, n)$  denote an optimal sequence of attribute levels for an instance of the lexicographic inference problem. Consider the feasible solution  $v_k = (n - k)v$  to the problem of maximizing  $E$ , where  $v > 0$  is a constant and  $k = 1, \dots, n$ . From Lemma 1, the probability of choosing the optimal sequence is given by

$$p(s^*) = p(u_1 > \dots > u_n) = \prod_{k=1}^{n-1} \frac{e^{v_k}}{e^{v_k} + \dots + e^{v_n}} \\ = \prod_{k=1}^{n-1} \frac{1}{1 + e^{v_{k+1}-v_k} + \dots + e^{v_n-v_k}}.$$

The exponents of the denominator terms in the last expression have the values

$$v_{k+j} - v_k = [n - (k + j)]v - (n - k)v = -jv, \text{ for all } 1 \leq j \leq n - k.$$

Thus,

$$p(s^*) = \prod_{k=1}^{n-1} \frac{1}{1 + e^{-v} + \dots + e^{-(n-k)v}}.$$

A lower bound on the value of  $p(s^*)$  is obtained by replacing each exponent  $-(n - j)v$  in the denominator by  $-v$ :

$$p(s^*) \geq \prod_{k=1}^{n-1} \frac{1}{1 + (n - k)e^{-v}}.$$

Let  $\epsilon > 0$  be any arbitrary value close to zero. Because the denominator on the right-hand side of this expression is a decreasing function of  $v$ , we can choose a value of  $v > 0$  for which

$$\prod_{k=1}^{n-1} \frac{1}{1 + (n - k)e^{-v}} \geq 1 - \epsilon.$$

It follows that

$$E = \sum_{s \in S} p(s)z(s) \geq p(s^*)z(s^*) > (1 - \epsilon)z^*,$$

where  $z(s) \geq 0$  is the number of nonreversals associated with the sequence  $s$ . On the other hand,  $E \leq z^*$  because the expected value of the number of nonreversals cannot exceed the maximum number of nonreversals across the sequences in  $S$ . Thus,

$$(1 - \epsilon)z^* < E \leq z^*.$$

As the value of  $v$  increases, the value of  $\epsilon$  decreases, and the lower and upper bounds in the preceding expression converge to  $E = z^*$ .  $\square$

**Proof of Lemma 2.** Consider the feasible solution  $v_j = v$  for all  $j = 1, \dots, n$ . Then, for any  $r = (i, h) \in R$ , the probability that

the highest ranked level in alternative  $i$  precedes the highest ranked level in alternative  $h$  has the value  $m_{1r}/(m_{1r} + m_{2r}) = 1/2$  because  $m_{1r} = m_{2r}$ . Because  $v_j = v$  is a feasible but not necessarily an optimal solution to the lexicographic inference problem,

$$z^* \geq \sum_{r \in R} \frac{m_{1r}}{m_{1r} + m_{2r}} = \frac{N}{2}. \quad \square$$

**Proof of Lemma 3.** Consider the feasible solution  $v_j = v$  for all  $j = 1, \dots, n$ . Then  $p(r) \geq 1/2$  for all  $r \in R$ . The corresponding value of the likelihood function is  $\mathcal{L} \geq 1/2^N$ . Because  $\mathcal{L}^*$  is the maximum value of  $\mathcal{L}$ ,  $\hat{\mathcal{L}} = (\mathcal{L}^*)^{1/N} \geq \mathcal{L}^{1/N} = 1/2$ .  $\square$

**Proof of Theorem 3.** We separately consider the two cases  $z^* = N$  and  $z^* \leq N - 1$ .

(1) Case 1:  $z^* = N$ . The maximum likelihood solution obtains the optimal ordering of the attribute levels, say  $1, \dots, n$ , when  $\hat{v}_j - \hat{v}_{j+1} = v$  for all  $j = 1, \dots, n - 1$ , where  $v > 0$  is an arbitrarily large number. It follows that  $\phi = \hat{a} = \hat{g} = 1$  when  $z^* = N$ .

(2) Case 2:  $z^* \leq N - 1$ . Because  $z^* \geq N\hat{a}$ , the lower bound in Lemma 4 gives

$$z^* \geq N\hat{a} \geq N\hat{g} + N \min \left\{ 2\sigma^2, \frac{\kappa^2\sigma^2}{\kappa^2 - \sigma^2} \right\}.$$

Thus,

$$\phi = \frac{N\hat{a}}{z^*} \geq K^* \left[ \hat{g} + \min \left\{ 2\sigma^2, \frac{\kappa^2\sigma^2}{\kappa^2 - \sigma^2} \right\} \right],$$

where  $K^* = N/z^* \geq 1$  and  $\hat{g} \geq 1/2$  from Lemma 3.  $\square$

## Appendix B. Overview of Hamiltonian Monte Carlo Algorithm

We provide a brief overview of the HMC algorithm. See Neal (1998) for details.

Let  $v = (v_1, \dots, v_n)$  denote the vector of deterministic utilities associated with the  $n$  attribute levels. The joint posterior density represents a well of potential energy. HMC assigns a fictitious momentum variable  $\mathcal{M}_k$  to each component  $v_k$  of  $v$ . All momentum variables are simultaneously updated in a Metropolis algorithm. The momentum vector determines the jumping distribution of  $v$ . A mass matrix,  $M$ , gives a weighting of the different components of the parameters vector. The momentum  $\mathcal{M}$  is sampled from a normal distribution  $\mathcal{N}(\mathcal{M}|v, M)$ . The values of  $v$  and the momentum are updated using the joint density  $p(v, \mathcal{M}|R) = p(v|R)p(\mathcal{M})$ , where  $R$  denotes the set of paired comparisons.

The momentum is only an auxiliary parameter, used to make the convergence of the algorithm faster. Updated draws of the parameters  $v$  are retained but the momentum updates are ignored. Practically, dynamics are incorporated by specifying a step size,  $\epsilon$ ; the mass matrix,  $M$ ; and a number of leapfrog steps,  $L$ , to simulate the dynamics of the sampler on the potential surface. Finding the right values of these auxiliary parameters can be difficult. We use the Stan probabilistic programming language (Hoffman and Gelman 2014) that incorporates a variant of HMC called the No U-turn (NUTS) sampler. In Stan, the auxiliary parameters are systematically and

automatically updated across the HMC iterations, allowing for a convergence of the Markov chain in fewer Markov Chain Monte Carlo iterations than the Metropolis–Hastings algorithm. More details of NUTS are provided by Hoffman and Gelman (2014). Algorithm 2 specifies the steps for a single HMC iteration.

**Algorithm 2** (HMC Iteration Given the Current Position  $v_{(0)}$ , Step Size  $\epsilon$ , Mass Matrix  $M$ , and Number of Leapfrog Steps  $L$ )

1. Initialize  $v_{(0)} \leftarrow v_{\text{current}}$
2. **For**  $l = 0, \dots, L - 1$ , **do** ▷ leapfrog steps
3.  $\mathcal{M}_{(l+1/2)} \leftarrow \mathcal{M}_{(l)} + \frac{1}{2}\epsilon \nabla_v \log p(v_{(l)}|y)$
4.  $v_{(l+1)} \leftarrow v_{(l)} + \epsilon M^{-1} \mathcal{M}_{(l+1/2)}$
5.  $\mathcal{M}_{(l+1)} \leftarrow \mathcal{M}_{(l+1/2)} + \frac{1}{2}\epsilon \nabla_v \log p(v_{(l+1)}|R)$
6. **End For**
7.  $\alpha = \min \left( 1, \frac{p(v_{(L)}|R)p(\mathcal{M}_{(L)})}{p(v_{(0)}|R)p(\mathcal{M}_{(0)})} \right)$  ▷ acceptance rate
8.  $u \sim \mathcal{U}(0, 1)$  ▷ uniform(0,1) draw
9. **Return**  $v_{(0)} + \delta_{u>\alpha}(v_{(L)} - v_{(0)})$  ▷  $\delta_{u>\alpha} = 1$  if  $u > \alpha$ ,  
0 otherwise

## Endnote

<sup>1</sup> The authors thank a reviewer for pointing out that generating paired comparisons from choice data has the limitation that it makes it impossible to identify certain effects, such as the decoy effect.

## References

- Ailon N, Charikar M, Newman A (2008) Aggregating inconsistent information: Ranking and clustering. *J. ACM* 55(5):1–27.
- Bartholdi J, Tovey CA, Trick MA (1989) Voting schemes for which it can be difficult to tell who won the election. *Soc. Choice Welfare* 6(2):157–165.
- Beggs S, Cardell S, Hausman J (1981) Assessing the potential demand for electric cars. *J. Econometrics* 17(1):1–19.
- Belloni A, Freund R, Selove M, Simester D (2008) Optimizing product line designs: Efficient methods and comparisons. *Management Sci.* 54(9):1544–1552.
- Bertsimas D, Mišić VV (2017) Robust product line design. *Oper. Res.* 65(1):19–37.
- Bertsimas D, Mišić VV (2019) Exact first-choice product line optimization. *Oper. Res.* Forthcoming.
- Bettman JR, Luce MF, Payne JW (1998) Constructive consumer choice processes. *J. Consumer Res.* 25(3):187–217.
- Billings RS, Scherer LL (1988) The effects of response mode and importance on decision-making strategies: Judgment versus choice. *Organ. Behav. Human Decision Processes* 41(1):1–19.
- Box GEP, Hunter WG, Hunter JS (1978) *Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building* (Wiley, New York).
- Bridges D (1983) A numerical representation of intransitive preferences on a countable set. *J. Econom. Theory* 30(1):213–217.
- Bröder A (2000) Assessing the empirical validity of the “take-the-best” heuristic as a model of human probabilistic inference. *J. Experiment. Psych. Learn. Memory Cognition* 26(5):1332–1346.
- Bröder A, Schiffer S (2003) Take the best versus simultaneous feature matching: Probabilistic inferences from memory and effects of representation format. *J. Experiment. Psych. General* 132(2):277–293.
- Byrd RH, Nocedal J, Schnabel RB (1994) Representations of quasi-Newton matrices and their use in limited memory methods. *Math. Programming* 63(4):129–156.
- Campos V, Glover F, Laguna M, Martí R (2001) An experimental evaluation of a scatter search for the linear ordering problem. *J. Global Optim.* 21(4):397–414.
- Charon I, Hudry O (2007) A survey on the linear ordering problem for weighted or unweighted tournaments. *4OR* 5(1):5–60.
- Charon I, Hudry O (2010) An updated survey on the linear ordering problem for weighted or unweighted tournaments. *Ann. Oper. Res.* 175(1):107–158.
- Chateaufneuf A (1987) Continuous representation of a preference relation on a connected topological space. *J. Math. Econom.* 16(2):139–146.
- Chen KD, Hausman WH (2000) Technical note: Mathematical properties of the optimal product line selection problem using choice-based conjoint analysis. *Management Sci.* 46(2):327–332.
- Debreu G (1954) Representation of a preference ordering by a numerical function. Thrall RM, Coombs CH, Davis RL, eds. *Decision Processes* (John Wiley, New York), 159–165.
- DeSarbo WS, Fong DKH, Liechty JC (2005) Dynamic models incorporating individual heterogeneity: Utility evolution in conjoint analysis. *Marketing Sci.* 24(2):285–293.
- Dieckmann A, Dippold K, Dietrich H (2009) Compensatory versus noncompensatory models for predicting consumer preferences. *Judgment Decision Making* 4(3):200–213.
- Drolet A, Luce MF (2004) Cognitive load and trade-off avoidance. *J. Consumer Res.* 31(1):63–77.
- Fishburn P (1974) Lexicographic orders, utilities and decision rules: A survey. *Management Sci.* 20(11):1442–1471.
- Floudas CA, Visweswaran V (1995) Quadratic optimization. Horst R, Pardalos PM, eds. *Handbook of Global Optimization*, Nonconvex Optimization and Its Applications, vol. 2 (Springer, Boston), 217–269.
- Garcia CG, Pérez-Brito D, Campos V, Martí R (2006) Variable neighborhood search for the linear ordering problem. *Comput. Oper. Res.* 33(12):3549–3565.
- Gigerenzer G, Hoffrage U, Kleinbolting H (1991) Probabilistic mental models: A Brunswikian theory of confidence. *Psych. Rev.* 98(4):506–528.
- Goemans MX, Williamson DP (1995) Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM* 42(6):1115–1145.
- Green PE, Krieger AM (1985) Models and heuristics for product line selection. *Marketing Sci.* 4(1):1–19.
- Grötschel M, Jünger M, Reinelt G (1984) A cutting plane algorithm for the linear ordering problem. *Oper. Res.* 32(6):1195–1220.
- Hoffman M, Gelman A (2014) The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Machine Learn. Res.* 15(1):1351–1381.
- Kemeny JG (1959) Mathematics without numbers. *Daedalus* 88(4):577–591.
- Knoblauch V (2000) Lexicographic orders and preference representation. *J. Math. Econom.* 34(2):255–267.
- Kohli R, Jedidi K (2007) Representation and inference of lexicographic preference structures and their variants. *Marketing Sci.* 26(3):380–399.
- Kohli R, Jedidi K (2015) Error theory for elimination by aspects. *Oper. Res.* 63(3):512–526.
- Kohli R, Krishnamurti R (1987) A heuristic approach to product design. *Management Sci.* 33(12):1523–1533.
- Laguna M, Martí R, Campos V (1999) Intensification and diversification with elite tabu search solutions for the linear ordering problem. *Comput. Oper. Res.* 26(12):1217–1230.
- Martí R, Reinelt G (2011) *The Linear Ordering Problem: Exact and Heuristic Methods in Combinatorial Optimization*, Applied Mathematical Sciences, vol. 175 (Springer-Verlag, Berlin, Heidelberg).
- Martignon L, Hoffrage U (1999) Why does one reason decision making work? Gigerenzer G, Todd PM, the ABC Research Group, eds. *Simple Heuristics That Make Us Smart* (Oxford University Press, New York), 119–140.
- Martignon L, Hoffrage U (2002) Fast, frugal and fit: Simple heuristics for paired comparison. *Theory Decision* 52(1):29–71.

- Motwani R, Raghavan P (1995) *Randomized Algorithms* (Cambridge University Press, New York).
- Neal RM (1998) Regression and classification using Gaussian process priors. Bernardo JM, Berger JO, Dawid AO, Smith AFM, eds. *Bayesian Statistics*, vol. 6 (Oxford University Press, New York), 475–501.
- Papadimitriou CH, Steiglitz K (1998) *Combinatorial Optimization: Algorithms and Complexity* (Dover Publications, New York).
- Payne JW (1982) Contingent decision behavior. *Psych. Bull.* 92(2): 382–402.
- Pinelis I (2015) Exact upper and lower bounds on the difference between the arithmetic and geometric means. *Bull. Australian Math. Soc.* 92(1):149–158.
- Schiavinotto T, Stützle T (2004) The linear ordering problem: Instances, search space analysis and algorithms. *J. Math. Model. Algorithms* 3(4):367–402.
- Schkade DA, Johnson EJ (1989) Cognitive processes in preference reversals. *Organ. Behav. Human Decision Processes* 44(2):203–231.
- Schmitt M, Martignon L (2006) On the complexity of learning lexicographic strategies. *J. Machine Learn.* 7(29):55–83.
- Slovic P (1975) Choice between equally valued alternatives. *J. Experiment. Psych. Human Perception Performance* 1(3):280–287.
- Toubia O, Hauser JR, Simester DI, Dahan E (2003) Fast polyhedral adaptive conjoint estimation. *Marketing Sci.* 22(3):273–303.
- Tversky A (1972) Choice by elimination. *J. Math. Psych.* 9(4):341–367.
- Tversky A, Sattath S, Slovic P (1988) Contingent weighting in judgment and choice. *Psych. Rev.* 95(3):371–384.
- van Zuylen A, Williamson DP (2008) Deterministic algorithms for rank aggregation and other ranking and clustering problems. Kaklamani C, Skutella M, eds. *Approximation and Online Algorithms. WAOA 2007. Lecture Notes in Computer Science*, vol. 4927 (Springer, Berlin, Heidelberg), 260–273.
- Wakker P (1988) Continuity of preference relations for separable topologies. *Internat. Econom. Rev.* 29(1):105–110.
- Yee M, Dahan E, Hauser JR, Orlin J (2007) Greedoid-based non-compensatory inference. *Marketing Sci.* 26(4):532–549.

---

**Rajeev Kohli** is the Ira Leon Rennert Professor of Business at the Graduate School of Business, Columbia University, New York. His research interests are in choice models, product design, combinatorial optimization, and algorithms.

**Khaled Boughanmi** is a PhD candidate in the marketing division at the Graduate School of Business, Columbia University, New York. His research interests are in the area of data-driven decision making, Bayesian nonparametrics, and machine learning with a focus on applications in e-commerce, personalization, online marketing, and service systems.

**Vikram Kohli** is an undergraduate computer science student in the McCormick School of Engineering at Northwestern University, Evanston, Illinois. His research interests are in machine learning, statistical natural language processing, artificial intelligence, and algorithms.