A Bias Correction Approach for Interference in Ranking Experiments

Ali Goli [*]
University of Washington

Anja Lambrecht
London Business School

Hema Yoganarasimhan
University of Washington

June 28, 2023

Abstract

Online marketplaces use ranking algorithms to determine the rank-ordering of items sold on their websites. The standard practice is to determine the optimal algorithm using A/B tests. We present a theoretical framework to characterize the Total Average Treatment Effect (TATE) of a ranking algorithm in an A/B test and show that naive TATE estimates can be biased due to interference. We propose a bias-correction approach that can recover the TATE of a ranking algorithm based on past A/B tests, even if those tests suffer from a combination of interference issues. Our solution leverages data across multiple experiments and identifies observations in partial equilibrium in each experiment, i.e., items close to their positions under the true counterfactual equilibrium of interest. We apply our framework to data from a travel website and present comprehensive evidence for interference bias in this setting. Next, we use our solution concept to build a customized deep learning model to predict the true TATE of the main algorithm of interest in our data. Counterfactual estimates from our model show that naive TATE estimates of clicks and bookings can be biased by as much as 15% and 28%, respectively.

Keywords: Experiments, A/B tests, Treatment Effects, Digital platforms, Interference, Machine Learning, Bias Correction

[*] We are grateful to an anonymous firm for providing the data. We thank Simha Mummalaneni and Omid Rafieian for detailed comments that have significantly improved the paper. Comments from the participants of 2021 INFORMS Marketing Science conference have helped the paper. We also thank seminar participants at Baruch College, Hong Kong Polytechnic University, and Yale University for their feedback. This research was supported by research funding from Marketing Science Institute. Please address all correspondence to: agoli@uw.edu, alambrecht@london.edu, hemay@uw.edu.

1 Introduction

Ranking algorithms play a central role in modern digital platforms and form the backbone of marketplaces such as AirBnB, Amazon, Expedia, Facebook, Instacart, TripAdvisor. These algorithms have a significant impact on consumers' responses (e.g., clicks and purchases) and platform's revenues since they determine the rank-ordering of products and services shown to consumers who visit these websites. Recognizing this, most modern platforms invest heavily in designing good ranking algorithms and evaluating them. The industry "gold standard" approach to evaluating ranking algorithms is A/B tests (or A/B/n tests in case of more than two conditions). This is typically done by allocating different portions of traffic to a set of competing ranking algorithms and then measuring each ranking algorithm's Total Average Treatment Effect (TATE) on outcomes such as the total number of clicks, sales, or revenue. Then, the ranking algorithm with the highest TATE in the A/B test is chosen as the winner and deployed for all users.

The inherent assumption underlying this practice is the Stable Unit Value Treatment Assumption (SUTVA) that assumes that the outcomes measured in a given treatment cell are independent of the treatments received by other cells (Imbens and Rubin, 2015; Rubin, 1990). Specifically, the assumption is that the TATE estimate of any given algorithm is independent of the mix of ranking algorithms that are being concurrently tested/deployed, and can thus be generalized to the population at large. However, this assumption often fails in marketplace settings involving A/B tests.

Indeed, a key challenge with A/B tests in marketplaces stems from the fact that outcomes observed under one treatment may depend on the treatments assigned to the rest of the population, leading to *interference bias* (Blake and Coey, 2014; Johari et al., 2020b; Bajari et al., 2020; Holtz et al., 2020). A particular worry in the context of testing alternative rankings is that something as simple as the share of the population assigned to different experimental conditions can bias the estimated TATEs (Ha-Thuc et al., 2020). This may be the result of, for example, supply-side constraints: a prominently ranked item[1] in the treatment condition may generate a high amount of activity or sales when the treatment group is comparatively small. However, this pattern may not hold when the treatment algorithm is applied to the full population if the item has limited availability, such as hotel rooms or flights (Johari et al., 2020b). Alternatively, ranking algorithms can have feedback loops whereby an item's ranking under one algorithm is directly affected by the total clicks or sales it receives (across all the experimental conditions). In such cases, the rank-ordering within a given test algorithm (and the experiment-based TATE) would end up being a function of the other algorithms in the experiment and the proportion of users assigned to them. Moreover, if the sellers (of the items) on the platform respond to the change prices or other supply-side variables in response to the observed demand, this can cause additional problems because the demand in the full implementation can diverge from the demand in the A/B test. In summary, marketplace experiments involving ranking algorithms can suffer from significant interference bias since they are unable to simulate what the *counterfactual market equilibrium* would be if the test algorithm were to be scaled up to the entire platform.

[1] Following, we use the term 'item' to refer to any type of units that may be listed by online merchants or platforms, including but not limited to products, services, suppliers, or search results.

Interference bias in A/B tests of ranking algorithms can have serious consequences for both inference and practice. From an inference standpoint, interference effects can severely bias the TATE estimates of the ranking algorithms derived from A/B tests. From a business perspective, such biased estimates can cause firms to pick sub-optimal ranking algorithms whose performance in A/B tests can (spuriously) appear to be more optimistic than their actual performance (when deployed on the entire platform). This, in turn, can have significant adverse effects on revenues, given the critical role that ranking algorithms play in these marketplaces. As such, developing a theoretically sound and practically viable solution to this problem is of primary importance to research and practice and the focus of this paper.

We establish the presence of interference bias in the TATE estimates of ranking algorithms based on A/B tests and provide a novel solution to this problem that can recover the TATE of a ranking algorithm based on past A/B tests, even if those tests suffer from a combination of interference issues. We then demonstrate the applicability of our approach using data from a major travel website. The first part of the paper consists of the theoretical framework and solution concept, while the second part focuses on the empirical application. We discuss each of these parts in detail below.

In the theoretical part of the paper, we first develop a framework to characterize the TATE of a given ranking algorithm (say a) under a A/B test. We write down a general demand model for each item sold by the platform under algorithm a. An important insight here is that an item's demand under algorithm a is not just a function of the distribution of rankings under a, but also the overall distribution of its rankings across all the algorithms that are being concurrently tested, i.e., the overall market equilibrium in the current regime (which is the A/B test). We show that this is due to interference from other ranking algorithms, e.g., the state variables associated with an item that affect its demand (e.g., price) within a are also influenced by other ranking algorithms. As such, when the item-level demand under algorithm a is aggregated over all the items to obtain an estimate of its TATE, this estimate is contaminated by the other algorithms in the experiment.

In general, when experiments suffer from interference issues or SUTVA failure, the standard solution is to re-design the experiment to avoid these types biases (e.g., block/cluster randomization, counterfactual ranking mechanisms). However, these solutions do not work in the case of ranking experiments because it is not possible for any A/B test to simulate what counterfactual market equilibrium would be when an algorithm a is fully deployed. (See §2 for a detailed discussion.) Therefore, we take a different approach in our solution. Instead of focusing on experiment design, we develop a procedure that takes data from a series of existing experiments and identifies micro-slices of data that are close to the counterfactual market equilibrium under a in each individual experiment. Thus, we can recover the true TATE of an algorithm a by aggregating these micro-slices of equilibrium outcomes across all items and experiments.

Specifically, we show that if the platform has access to multiple cohorts of experiments (i.e., multiple A/B tests) involving the algorithm of interest, then in each cohort, it is possible to observe a partial equilibrium, where some items are close to their equilibrium positions induced by a. As long as the number of experiments and the scope of the data is sufficiently large, pooling these observations across multiple cohorts will allow us to infer the TATE of a on the entire population, even if no one experiment was ideal. We operationalize this idea using the concept of *rank discrepancy*, which captures the difference between an item's rank within algorithm a and its overall rank in the cohort. Items or observations with low discrepancy can be interpreted as being close to their true market-equilibrium under a, and can contribute to the identification of the true TATE. Further, we show that even if some items always have high discrepancy across all cohorts, we can still recover their equilibrium outcomes under a if there are one or more items *similar* to j but have low discrepancy in some cohorts. Thus, our solution concept exploits two types of variations in the data – (1) variation in rank discrepancy of the same item across different cohorts and (2) variation in rank discrepancy for similar types of items. In summary, by isolating and pooling observations that have low rank discrepancy, we can identify the performance of an algorithm when it is scaled-up to the entire platform. As such, our solution requires the platform to have data from multiple cohorts of A/B tests to be practically useful. However, this requirement is relatively easy to satisfy in most modern platforms that have continuously running experimentation systems.

We apply our method to data from a large travel website specializing in hotel bookings. Users on this platform search or query for hotels by location and travel dates and are shown a list of hotels in response. Users can then click on individual offerings and, potentially, book a hotel. The firm routinely experiments with different ranking algorithms, and each user who visits the platform on any given day is randomly assigned one of the ranking algorithms being tested that day. During our observation period, one of the main algorithms of interest for the platform, "Genesis", was tested against different sets of algorithms over a series of experiments. The share of users assigned to Genesis varied widely across experiments, but it was never deployed at 100%. Our goal is to obtain a good estimate of Genesis's performance if it were to be deployed to all users on the platform.

Our empirical analysis consists of two parts. In the first part, we present evidence for interference bias in our setting using a series of descriptive analyses and Difference-in-Difference (DiD) models. First, we show that Genesis's performance varies with the share of the population assigned to it, which should not happen if there is no interference bias. Next, we focus on the last two cohorts/experiments in our data and document a discontinuous drop in the performance of Genesis across the two cohorts. We show that this discontinuity can be attributed to interference bias rather than other platform- or demand-related changes. In the second part of our empirical analysis, we develop a customized machine learning model to predict the true TATE of an algorithm. Our method builds on the theoretical solution described earlier and consists of four steps. First, we define our TATE estimation task, which requires us to predict the demand for each hotel on a given day under Genesis as a function of hotel features, daily seasonality features, and a set of ranking discrepancy features. In the second step, we develop the feature sets. The first two feature vectors are designed using representation learning methods, e.g., GloVE (Pennington et al., 2014). We use variables that capture the similarity between the hotel's ranking under Genesis and its overall ranking in that experiment for the discrepancy features. In the third step, we train three feed-forward neural networks to predict the three equilibrium outcomes: the number of daily impressions of a hotel given queries, number of clicks given impressions, and number of bookings given clicks. In the fourth step, we establish that our data satisfy the identification requirements necessary to apply our solution concept.

Finally, we perform the counterfactual analysis where we estimate the true TATE of the Genesis algorithm by setting the discrepancy measures to zero and aggregating the demand over all the hotels. We find that our estimate of the true TATE is systematically different from the naive TATE measurements from the A/B tests and is a function of the extent of interference in a given experiment. Following up on the reduced-form analysis, we zoom in on the last two cohorts. Unlike the naive TATE measurements, which drop discontinuously across the last two consecutive cohorts due to interference bias, the estimated true TATE shifts continuously. Further, the relative bias in observed vs. true (predicted) TATE shrinks from 15.2% to -3.4% as the Genesis algorithm is scaled up significantly in the last cohort. Next, we quantify the economic impact of interference by calculating the relative bias in the TATE measurements of bookings. We find that the relative bias is further inflated in the case of bookings, e.g., 28.5% and 10.5% in the last two cohorts. Together, these findings suggest that naively relying on simple A/B tests to measure the performance of ranking algorithms can lead to incorrect inference and sub-optimal decisions.

Our paper makes three contributions to the literature and practice of implementing and interpreting experiments involving ranking algorithms. First, in the context of testing ranking algorithms, we theoretically establish that the outcomes under one algorithm can be directly dependent on other algorithms being concurrently tested (in violation of SUTVA). Second, we propose a bias correction approach to recover the true TATE estimates of ranking algorithms using data from a series of A/B tests. Our method is agnostic to the source of interference and therefore applicable to a broad range of settings. An important feature of our approach is that it can be used with data from standard A/B tests available to firms and does not require firms to change their testing infrastructure. Third, we empirically quantify the magnitude of interference bias in a large travel platform and demonstrate how managers can apply our method to their settings. We expect our theoretical framework and empirical approach to be practically useful to firms interested in measuring the TATE of ranking algorithms.

2 Related Literature

Our work relates to the growing literature on the challenges associated with the design, implementation, and interpretation of the findings from large-scale experiments. Past papers in this stream have covered a wide variety of experimentation-related issues and proposed solutions to help firms and researchers avoid biased estimates of treatment effects, such as recognizing and accounting for early stopping and p-hacking (Johari et al., 2020b; Jamieson and Jain, 2018) and accounting for the failure of SUTVA in experiments on social networks and interference due to network effects (Manski, 2013; Saveski et al., 2017; Eckles et al., 2017; Nandy et al., 2019). Chapter 22 of Kohavi et al. (2020) provides an excellent discussion of these issues and the solutions available.

Our work is most directly related to prior research on interference bias in experiments on two-sided markets and ranking algorithms. Early papers in this area focused on documenting the presence of interference bias in two-sided markets. Blake and Coey (2014) use data from an email marketing campaign by eBay and show that interference among bidders can lead to misestimation of treatment effects by a factor of two. This happens because the emails induce test users to win more auctions but also cause the control users to lose with a higher probability. Further, in the context of ranking algorithms, Fradkin (2019) uses simulations to show that naive analysis of experiments involving ranking algorithms can overestimate conversion rates by 53% (when a quarter of users are randomized into treatment). Recent papers try to address interference, and the solutions they provide are typically specific to the context where such bias arises. For two-sided markets, Johari et al. (2020a) use a theoretical model to characterize the size and magnitude of the bias and propose a double randomization design similar to that suggested by Bajari et al. (2020), whereby randomization is done on both sides of the market (e.g. a pair of user/property). Another common approach to mitigate interference bias in two-sided markets is to employ block-randomization (Holtz and Aral, 2020; Holtz et al., 2020). More closely related to our context, Ha-Thuc et al. (2020) study interference bias in seller-side A/B tests of ranking in seller-side recommendation systems (e.g., boosting new sellers). They present a framework for seller-side testing that uses the counterfactual rank that the seller would have been assigned under the fully-scaled up version of the algorithm being tested instead of the the rank assigned under the experiment (where only a small percentage of users are assigned to the algorithm).

However, none of the above approaches translate well to a setting where the platform seeks to test the performance of different ranking algorithms in the presence of supply-side effects. The double-randomization method is only applicable at the item level and is not feasible at the list level (i.e., list of results as in the ranking case). While cluster-randomization is possible in principle, it requires both a sufficient number of non-interfering clusters and a sufficient number of observations in each cluster, which is practically infeasible in many settings (including ours). Finally, the approach by Ha-Thuc et al. (2020) only works when the interference stems from the platform's ranking algorithm. It cannot simulate counterfactual rankings when we also have interference due to market equilibrium effects, or supply-side responses and constraints, e.g., through prices and product availability in a two-sided market.

Our approach contributes to this literature by providing a method that can be applied to data from experiments on ranking algorithms where randomization was done at the user level. Unlike the previous papers, our approach is agnostic to the specific mechanism that leads to interference and can account for market-equilibrium effects. Further, from a firm's perspective, a practical advantage is that the proposed approach does not require them to employ specific randomization strategies or alter their experimentation infrastructure. Instead, firms can simply use pre-existing experimental data.

Finally, our paper relates to the literature on Learning to Rank, i.e., how to design optimal ranking algorithms in both computer science and marketing (Falk, 2019; Liu et al., 2021; Yoganarasimhan, 2020). Our work is also directly relevant to the growing literature on the evaluation of ranking algorithms and recommendation systems. Business use different methods to evaluate the performance of recommendation systems, including evaluations based on logged data or offline experiments (Joachims et al., 2017; Gomez-Uribe and Hunt, 2015), A/B tests (Chen and Canny, 2011; Katukuri et al., 2014), and interleaved experiments (Chapelle et al., 2012). Jannach and Jugovac (2019) provide a comprehensive review on the methods available to evaluate the performance of recommendation systems and discuss pitfalls that could bias the results.[2]

[2] A separate stream of work in economics and marketing seeks to measure position effects (Ghose et al., 2014; De los Santos and Koulayev, 2017; Ursu, 2018), where the goal is to measure the causal impact of an item's position on its demand. In contrast, our

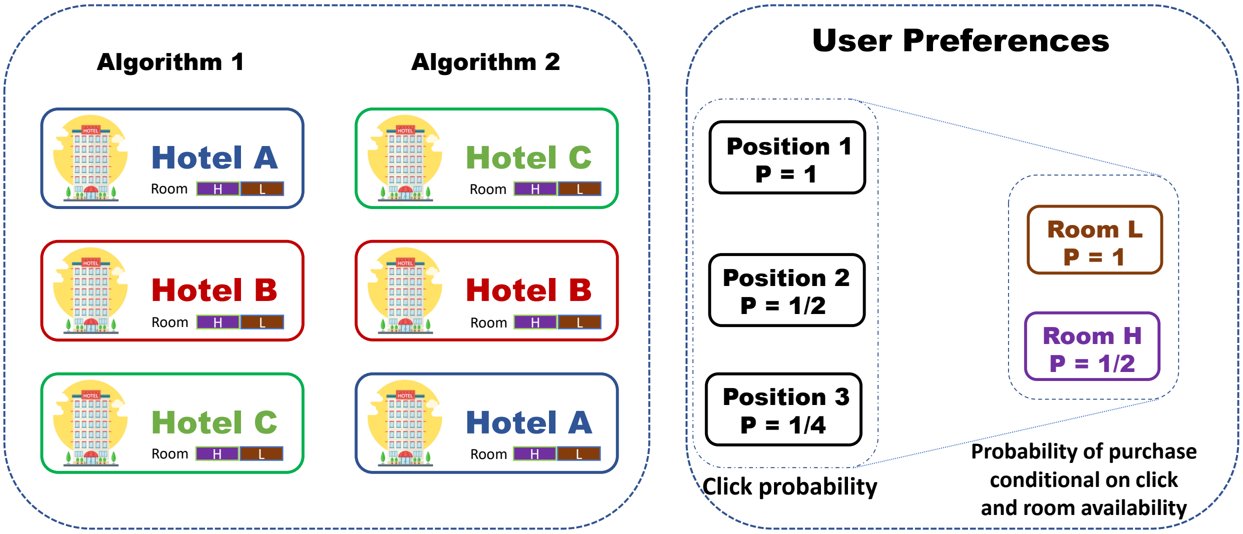


Figure 1: A pictorial illustration of the simple example. Notice that hotel B is ranked similarly in both algorithms 1 and 2, however, hotels A, and C are ranked differently by the two algorithm. Each hotel has only two identical rooms, with high (H) and low (L) price. Conditional on considering a hotel, if the low price listing is available users purchase it with probability 1, and if the low price room is already booked, the high price room is purchased with probability $\frac{1}{2}$. User preference, probability to consider a hotel, are 1, $\frac{1}{2}$, $\frac{1}{4}$, for positions 1, 2, and 3, respectively.

3 Interference Bias in Ranking Experiments

We now discuss how experiments on ranking algorithms can suffer from interference bias. To fix ideas, we start with a simple example in §3.1. Then, in §3.2, we discuss other potential sources of interference and how they can bias inference.

3.1 A Simple Example

Consider a hotel booking platform that has developed two algorithms for ranking hotels on its website. The goal is to test these algorithms before fully deploying one of them on the platform. For simplicity, assume that the platform has two users (U_1 and U_2) and three hotels, namely, A, B, and C. The first algorithm ranks the hotels in the order $A > B > C$, whereas the second algorithm ranks them as $C > B > A$. Since we have only two users, let us also assume that each hotel has two rooms, with a low (L) and a high price (H). Let a user's probability of considering a hotel in positions 1, 2, and 3 be equal to 1, 1/2, and 1/4, respectively. Users prefer the low-priced room to the higher-priced one, and if the low-priced one is sold out, the probability of purchase declines from 1 to 1/2. Figure 1 illustrates the rankings by each algorithm and users' preferences.[3]

goal is slightly different and more in line with that of the ranking literature, which is to reliably measure the TATE of a ranking algorithm in a market setting.

[3] In this simple model, all the hotels are homogeneous, and the heterogeneity in realized demand across hotels is purely due to differences in their ranking and room prices.

Now we can calculate the total demand for Hotel A when *only algorithm 1* is deployed on the platform as:

$$\begin{aligned}\mathbb{E}_1[D_A] &= \mathbb{E}_1[D_A \mid U_1 \text{ purchases A}] \cdot \mathbb{P}(U_1 \text{ purchases A}) \\ &+ \mathbb{E}_1[D_A \mid U_1 \text{ does not purchase A}] \cdot \mathbb{P}(U_1 \text{ does not purchase A}), \\ &= (1+\frac{1}{2}) \cdot 1 + (0+1) \cdot 0 = \frac{3}{2},\end{aligned} \tag{1}$$

where $\mathbb{E}_1[D_A \mid U_1 \text{ purchases A}]$ is the total demand for A (from both users) when U_1 has purchased A. Under this condition, the expected demand generated from U_1 is equal to 1 and the expected demand from U_2 is equal to $\frac{1}{2}$, since the L type room is purchased by U_1 when U_2 arrives at the platform. Therefore, the total expected demand for hotel A would be $1 + \frac{1}{2}$ and the average demand per user for hotel A when only algorithm 1 deployed on the platform is $\frac{3}{4}$. We perform similar calculations for hotels B and C and show the results in the first column of Table 1. The right panel of Figure 2 illustrates the steps for the calculations.

Now consider a scenario where the firm runs an A/B test and randomly assigns one customer to algorithm 1 and the other customer to algorithm 2. If the first customer, U_1, is assigned to algorithm 1 then the second one is assigned to algorithm 2, and vice versa. These events take place with probability $\frac{1}{2}$. Then the expected demand of Hotel A per user under algorithm 1 is equal to:

$$\begin{aligned}\mathbb{E}_1[D_A] &= \mathbb{E}_1[D_A \mid U_1 \text{ assigned to algorithm 1}] \cdot \mathbb{P}(U_1 \text{ assigned to algorithm 1}) \\ &+ \mathbb{E}_1[D_A \mid U_2 \text{ assigned to algorithm 1}] \cdot \mathbb{P}(U_2 \text{ assigned to algorithm 1}) \\ &= 1 \cdot \frac{1}{2} + \left(\frac{1}{4} \cdot \frac{1}{2} + \frac{3}{4} \cdot 1\right) \cdot \frac{1}{2} = \frac{15}{16}.\end{aligned} \tag{2}$$

The left panel of Figure 2 illustrates the steps for deriving Equation (2). We perform similar calculations for hotels B and C and show the results in the second column of Table 1. Comparing in Columns (1) and (2) of Table 1 the average demand generated (per user) when algorithm 1 is fully deployed with average demand generated by users assigned to algorithm 1 under the A/B test, we find that the demand for (1) hotel A is overestimated, (2) hotel B is correctly estimated, and (3) hotel C is underestimated. Note the ranking for hotel B is consistent across both algorithms, and this helped us recover the correct demand for hotel B when algorithm 1 is scaled up. However, when algorithms 1 and 2 are deployed concurrently, the discrepancy between the rankings for hotels A and C and the hotels' supply constraints in low-priced rooms imply that the prices users face under algorithm 1 in the A/B test are systematically different from the prices they would see when algorithm 1 is fully scaled up. In particular, for hotel A, its ranking under algorithm 2 is lower than its ranking under algorithm 1. Thus, the price distribution faced by users during the A/B test is lower than the price distribution they would face when algorithm 1 is fully deployed. In contrast, for Hotel C, the price distribution appears to be higher under the A/B test than it would be when algorithm 1 is fully deployed.

Finally, the total predicted demand per user for algorithm 1 based on the experiment is $\frac{25}{16} = 1.535$.[4] This is the *observed* Total Average Treatment Effect (TATE) of algorithm 1 from the experiment. In contrast, when

[4] For simplicity, in the example, we do not constrain the user to purchase only one item; so demand per user can be greater than one.

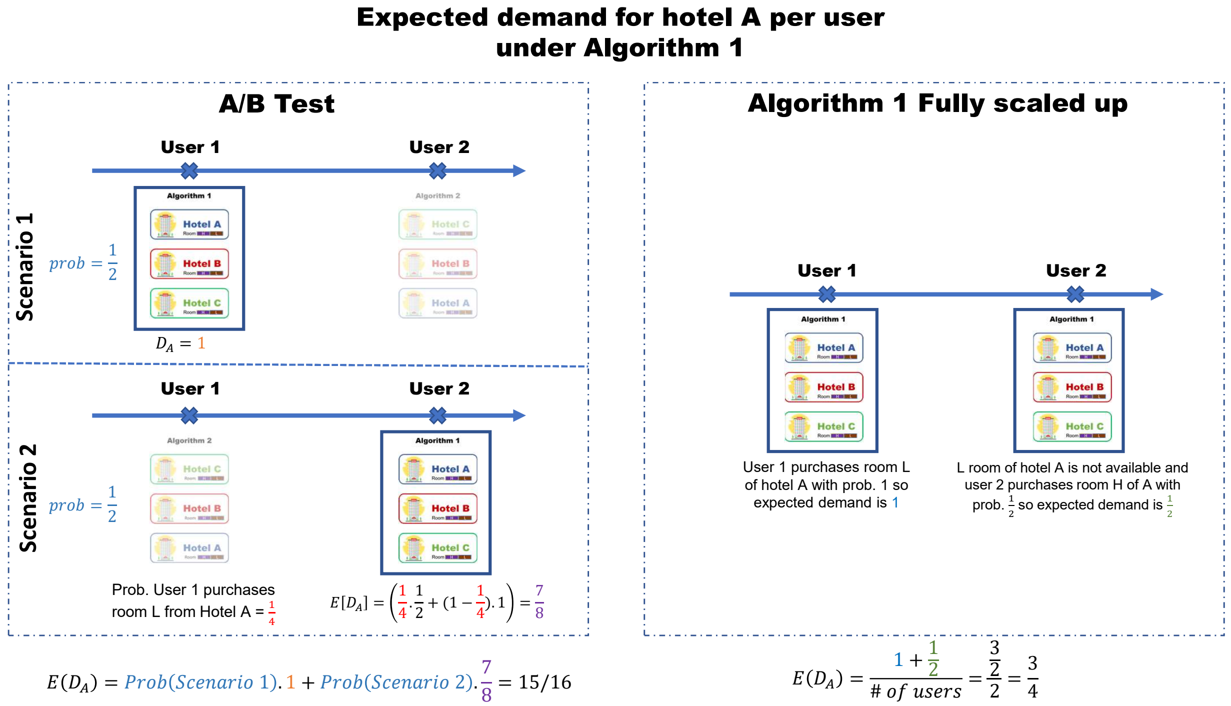

Expected demand for hotel A per user
under Algorithm 1
A/B Test
Scenario 1
User 1
User 2
prob = 1/2
Algorithm 1
Hotel A
Hotel B
Hotel C
Algorithm 2
Hotel C
Hotel B
Hotel A
D_A = 1
Scenario 2
User 1
User 2
prob = 1/2
Algorithm 2
Hotel C
Hotel B
Hotel A
Algorithm 1
Hotel A
Hotel B
Hotel C
Prob. User 1 purchases room L from Hotel A = 1/4
E[D_A] = (1/4 . 1/2 + (1 − 1/4). 1) = 7/8
E(D_A) = Prob(Scenario 1). 1 + Prob(Scenario 2). 7/8 = 15/16
Algorithm 1 Fully scaled up
User 1
User 2
Algorithm 1
Hotel A
Hotel B
Hotel C
Algorithm 1
Hotel A
Hotel B
Hotel C
User 1 purchases room L of hotel A with prob. 1 so expected demand is 1
L room of hotel A is not available and user 2 purchases room H of A with prob. 1/2 so expected demand is 1/2
E(D_A) = (1 + 1/2) / # of users = (3/2)/2 = 3/4

Figure 2: The left panel illustrates the process for deriving the expected demand per user for hotel A under the A/B test. We consider two scenarios depending on the order in which users are assigned to algorithm 1. First, calculate the expected demand for hotel A conditional on U_1 being assigned to algorithm 1: 1 (probability of U_1 to click on hotel A) x 1 (probability of purchasing conditional on clicking). Multiply this term with the probability of 1/2 that U_1 is indeed assigned to algorithm 1. Second, calculate the demand if U_2 is assigned to algorithm 1 and U_1 is assigned to algorithm 2. Recall that U_1 goes first. The probability that the less expensive room is no longer available for U_2 equals the probability that U_1 has chosen the less expensive room: 1/4 (probability that U_1 clicks on hotel A that is now in position 3 for that user) x 1 (probability that U_1 purchases hotel A conditional on clicking). Multiply this with 1 (probability for U_2 to click on hotel A since this is now in position 1) x 1/2 (probability of purchasing the available more expensive room conditional on clicking). In turn, the probability that user 2 chooses the less expensive room is 3/4 (probability that U_1 has not purchased hotel A) x 1 (probability for U_2 to click on hotel A) x 1 (probability to purchase the less expensive room conditional on clicking). Again, we multiply this term with with the probability of 1/2 that U_1 is indeed assigned to algorithm 2. The steps for calculating the per user demand under algorithm A when fully scaled up are illustrated in the right panel.

we deploy algorithm 1 on the entire platform, the total expected demand per user is $\frac{91}{64} = 1.422$, which is the *true* TATE of algorithm 1. Thus, if we naively use the observed TATE from the experiment, we over-estimate the effectiveness of algorithm 1 because it fails to account for price changes and supply-side constraints.

There are two main takeaways from this exercise. First, price response due to demand changes can distort estimates of treatment effects of ranking algorithms obtained from naive A/B tests. Second, the bias in our prediction of a hotel's demand under a given algorithm is a function of the discrepancy in the hotel's average ranking in the experiment relative to its counterfactual ranking when the algorithm is fully scaled up. This last point is an important insight that we will leverage when we formulate our solution and empirical approach.

3.2 Other Sources of Interference Bias

The example in the previous section highlighted one source of interference. In practice, there are many other sources of interference that can hamper our ability to correctly infer the effectiveness of ranking algorithms.

	Expected demand per user when algorithm 1 is fully scaled	Expected demand per user assigned to algorithm 1 under the A/B test
Hotel A	$\frac{3}{4} = 0.75$	$\frac{15}{16} = 0.94$
Hotel B	$\frac{7}{16} = 0.44$	$\frac{7}{16} = 0.44$
Hotel C	$\frac{15}{64} = 0.23$	$\frac{3}{16} = 0.19$
Total demand per user	$\frac{91}{64} = 1.42$	$\frac{25}{16} = 1.56$

Table 1: Expected demand per user for each hotel under algorithm 1 measured when both users are assigned to algorithm 1 (left column) and under the A/B test (right column).

For instance, if an item's price changes in response to its overall demand (even in the absence of capacity constraints), equilibrium prices under the A/B test would be systematically different when the algorithm is fully scaled up. This, in turn, can lead to misleading estimates of an algorithm's performance. In fact, any item-specific state variable that could change as a function of aggregate demand that is factored into consumer choices (e.g., promotions, advertising) could lead to bias in our measure of the algorithm's performance. We refer to these types of interference effects as *indirect interference*, wherein the other algorithms in the A/B test interfere with outcomes in the focal algorithm only through these endogenous state variables.

Interference bias can also occur if a focal ranking algorithm uses outcomes from other algorithms as inputs into its ranking function (even in the absence of supply-side response or constraints). Consider, for instance, algorithm 1, which sorts hotels based on popularity, measured as the total number of bookings made for a given hotel; that is, hotels with more overall bookings are placed in a higher position. Now suppose that this algorithm is tested against an algorithm 2 that has a fixed ranking for each hotel. Then the popularity ranking of a hotel in algorithm 1 can change depending on the portion of traffic assigned to algorithm 2. This, in turn, would make the rankings of algorithm 1 endogenous to the experiment, i.e., a function of the mix of algorithms being tested. As such, the inference from an A/B test will be contaminated by other algorithms being concurrently tested, and we cannot correctly predict the performance of algorithm 1 when it is fully deployed. We refer to this type of interference as *direct interference*, wherein the algorithm of interest is directly influenced by outcomes in other algorithms in the A/B test. (In §4, we formally define these two notions of interference and explain how our solution concept can accommodate both of them.)

Further, the direction of the interference bias can be either positive or negative, depending on whether the treatment effects of individual items are over-or underestimated. So there is no theoretical guidance on how to sign the bias. In this paper, we are agnostic to the exact source of interference and do not seek to pin down or focus on a specific source of interference. Indeed, any or all the above sources of interference can be at play in large platforms. Instead, our goal is to propose a solution that can recover the TATE of a specific algorithm based on past A/B tests, even if those tests suffer from a combination of interference issues.

4 Model and Solution Concept

In §4.1, we present a general model to characterize the estimated demand or the TATE of a ranking algorithm measured based on data from an A/B test. In §4.2, we show how TATE measurements based on A/B tests diverge from the true TATE. Next, in §4.3, we present our solution to recover the true TATE based on data from a series of A/B tests.

4.1 Demand Model

Consider a platform that has a unit mass of users that are indexed by i. Each user i who visits the platform is randomly assigned to one of the algorithms $a_i \in \mathcal{A}$, where $\mathcal{A}$ is the set of all ranking algorithms available on the platform. The mass of customers assigned to algorithm a is equal to m_a and $\sum_{a \in \mathcal{A}} m_a = 1$. Also, let $\mathbf{m} = (m_1, \ldots, m_{|\mathcal{A}|})$ be a vector that reflects what portion of customers are assigned to each algorithm in this regime. When customer i, who is assigned to algorithm a, submits a query, she is shown a rank-ordered list of items based on a. We assume that the rank r for an item j under algorithm a follows a distribution with a probability mass function on natural numbers that is denoted by $P_j^a(r)$.[5] Upon seeing this list, user i can click and buy one or more of the items listed.

For expositional simplicity, we start our analysis with the simple case where there is only *indirect interference*. Mathematically, when there is *no direct interference*, the distribution of rankings for an item j under a given algorithm a does not depend on the mix of algorithms that are being tested, i.e., $P_j^a \perp \mathbf{m}$. So the rank distribution for a given hotel j under algorithm a, $P_j^a(r)$, remains the same regardless of $\mathbf{m}$. We first discuss the model formulation and solution concept under this assumption. At the end of this section, we consider the more general case with both direct and indirect interference.

We now construct a model for customer i's demand for item j when the rank-ordered listings for customer i are generated using algorithm a. We consider a two-step procedure for constructing the demand function. In the first step, we assume that the probability of an item j being considered by user i is only a function of item j's ranking by algorithm a, and is equal to γ_r for an item of rank r. In the second step, the customer decides to click and purchase a subset of items that are in her consideration set.[6] Here, we assume that the probability of purchase conditional on consideration is independent of the item's rank. This assumption is not required for our method and is not used in our solution concept. However, we use it to keep the discussion accessible and consistent with the literature on counterfactual evaluation of ranking algorithms (see Joachims et al. (2017) and Ha-Thuc et al. (2020)).

Once an item is considered, the user purchases the item with probability $\alpha(\mathbf{x}_j, \mathcal{S}_j)$, where $\mathbf{x}_j$ and $\mathcal{S}_j$ are item-fixed characteristics and state for item j, respectively.[7] The vector $\mathbf{x}_j$ consists of time-invariant features of the item that cannot change as a function of the market equilibrium (e.g., location of a hotel or the amenities available). The vector $\mathcal{S}_j$ represents time-varying features of the item that could change as a

[5] The ranking of a given item in an algorithm need not be deterministic. In the case of hotel rankings on a travel website, this could happen because the hotel appears in different positions when the query changes. For instance, a hotel may appear in the results both when the query is "London" and "Central London" albeit at different positions. Another possibility is that the position could shift as the mix of available hotels could change when the arrival dates or number of nights change.

[6] To simplify matters, here we do not separately model the process of clicking on impressions and booking. Later in §7, we describe the different steps in detail.

[7] Notice that conditional on consideration, a product's purchase probability is only a function of its own attributes and not that of other items in the list. This is a standard assumption in the ranking literature (Joachims et al., 2017) and is similar in spirit to the "no local interference" assumption in non-ranking settings (Bajari et al., 2020).

function of overall demand; for example, in our empirical application, this could include variables such as the hotel's prices or the mix of rooms that are available. Then, the expected probability of a user assigned to algorithm a purchasing item j, given state $\mathcal{S}_j$, is:

$$q_a^{(d)}(\mathbf{x}_j, \mathcal{S}_j; P_j^a(\cdot)) = \mathbb{E}_r \gamma_r \alpha(\mathbf{x}_j, \mathcal{S}_j) = \sum_{r=1}^{\infty} \gamma_r \alpha(\mathbf{x}_j, \mathcal{S}_j) P_j^a(r), \tag{3}$$

where the expectation is taken over the rank assigned by algorithm a to listing j. The overall demand for item j is then calculated by aggregating the demand generated from all the users assigned to all the algorithms employed in the experiment, as follows:

$$\begin{aligned} q^{(d)}(\mathbf{x}_j, \mathcal{S}_j; P_j^a(\cdot), \mathbf{m}, \mathcal{A}) &= \sum_{a \in \mathcal{A}} m_a q_a^{(d)}(\mathbf{x}_j, \mathcal{S}_j) = \sum_{r=1}^{\infty} \sum_{a \in \mathcal{A}} m_a \gamma_r \alpha(\mathbf{x}_j, \mathcal{S}_j) P_j^a(r) \\ &= \sum_{r=1}^{\infty} \underbrace{\sum_{a \in \mathcal{A}} m_a P_j^a(r)}_{P_j^{\mathbf{m}}(r)} \gamma_r \alpha(\mathbf{x}_j, \mathcal{S}_j) = q^{(d)}(\mathbf{x}_j, \mathcal{S}_j; P_j^{\mathbf{m}}(\cdot)), \end{aligned} \tag{4}$$

where $P_j^{\mathbf{m}}(r)$ is the probability mass function of rankings for item j given the set of algorithms $\mathcal{A}$ assigned according to the probability vector $\mathbf{m}$. Note that the overall demand for each item depends on the probability mass function of the ranking for that item by the mix of algorithms $\mathbf{m}$, $P_j^{\mathbf{m}}(.)$, which can be conveyed fully by the information in $\mathbf{m}$. In other words, we do not need to know the distribution of rankings of an item under all the algorithms to calculate the overall demand; rather, the overall ranking of an item by the mix of algorithms contains the necessary information to calculate demand.

For the supply side, the quantity supplied by each item is a function of its characteristics $\mathbf{x}_j$ and state $\mathcal{S}_j$ and is denoted by $q^{(s)}(\mathbf{x}_j, \mathcal{S}_j)$. In equilibrium, for each item j we must have $q^{(s)}(\mathbf{x}_j, \mathcal{S}_j) = q^{(d)}(\mathbf{x}_j, \mathcal{S}_j; P_j^{\mathbf{m}}(\cdot))$, which implies that the item-state ($\mathcal{S}_j$) faced by customers under algorithm a is also a function of the other algorithms in the experiment since the probability mass function $P_j^{\mathbf{m}}(.)$ is generated by the mix of ranking algorithms $\mathbf{m} = (m_1, \ldots, m_{|\mathcal{A}|})$. The state $\mathcal{S}_j$ is thus an equilibrium outcome. Therefore, the demand for item j generated by users exposed to algorithm a not only depends on how algorithm a ranks j but also on the expected overall ranking $P_j^{\mathbf{m}}(.)$ for item j since the latter affects the equilibrium state for item j. To proceed with the analysis, we need to make the following regularity assumption on $\mathcal{S}_j$.

Assumption 1. *An item j's state, $\mathcal{S}_j$, is a unique and continuous function of the overall ranking distribution $P_j^{\mathbf{m}}(.)$, and $\mathbf{x}_j$.*

This assumption states that the state variables associated with j (e.g., prices and distribution of rooms in the case of hotel rooms) change smoothly and can be described as a function of item characteristics and its position in the cohort. Based on Assumption 1, we know that the state of item j in equilibrium only depends on $P_j^{\mathbf{m}}(\cdot)$ and $\mathbf{x}_j$. Therefore, $\{\mathbf{x}_j, P_j^a(\cdot), P_j^{\mathbf{m}}(\cdot))\}$ contain all the information necessary to capture $\mathcal{S}_j$, i.e., $q_a^{(d)}(\mathbf{x}_j, \mathcal{S}_j; P_j^a(\cdot), P_j^{\mathbf{m}}(\cdot)) = q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}}(\cdot))$. For notational and expositional simplicity, we do not explicitly include exogenous factors that can affect both the equilibrium state of an item and its demand (e.g., demand seasonality or changes in the composition of users over time) in our theoretical framework. However, these factors are incorporated into the empirical solution and discussed in detail in §7.

4.2 Total Average Treatment Effect

In the above section, we showed that an item's state $\mathcal{S}_j$ is a function of the *overall* demand for that item and that it can change as a function of the mix of algorithms that are being tested on the platform. Further, because this state affects demand through consumer's purchase probabilities ($\alpha(\mathbf{x}_j, \mathcal{S}_j)$), the observed demand for an item j is a function of the overall experiment. We now discuss the implications of these findings for TATE estimation.

Recall that our goal is to infer the performance of algorithm a when all users are assigned to it. In other words, our goal is to learn the demand generated for an item j shown in algorithm a, $q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot))$. The true Total Average Treatment Effect (TATE) of the algorithm, which is the total demand generated per user under algorithm a when it is fully scaled up, is equal to:

$$\text{True TATE} = \sum_{j \in J} q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot)). \tag{5}$$

However, our data comes from an experiment where only m_a portion of the users were assigned to algorithm a, and the vector of treatment assignment is given by $\mathbf{m}$. Recall that the state of the item j, $\mathcal{S}_j$, is a function of the *overall* demand for item j, which depends on the distribution of overall rankings $P_j^{\mathbf{m}}(\cdot)$. Thus, the TATE observed in the experiment is equal to:

$$\text{Observed TATE} = \sum_{j \in J} q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}}(\cdot)). \tag{6}$$

As such, unless $P_j^{\mathbf{m}}(\cdot) = P_j^a(\cdot)$, we would expect the TATE observed in the experiment to be a biased estimate of the true TATE. Thus, the extent to which the estimated TATE, $\sum_{j \in J} q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}}(\cdot))$, diverges from the true TATE, $\sum_{j \in J} q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot))$, depends on the discrepancy between $P_j^a(\cdot)$ and $P_j^{\mathbf{m}}(\cdot)$. We now build on this insight and propose a solution to infer the true TATE.

4.3 Solution

Before elaborating our solution concept, we first define the notion of a cohort as follows.

Definition 1. A cohort c is defined as a period where a set of algorithms are tested in fixed proportions (denoted by the vector $\mathbf{m}_c$) against each other using a randomized control trial.

Continuous experimentation is standard in most modern platforms, and each cohort or experiment is often followed by another cohort that tests a new set of algorithms, often with some algorithms running continuously to test against others.[8]

[8] For ease of exposition, we assume that all algorithms in a cohort have the same defined start and end point, but our method generalizes to contexts when each algorithm starts and ends at different times.

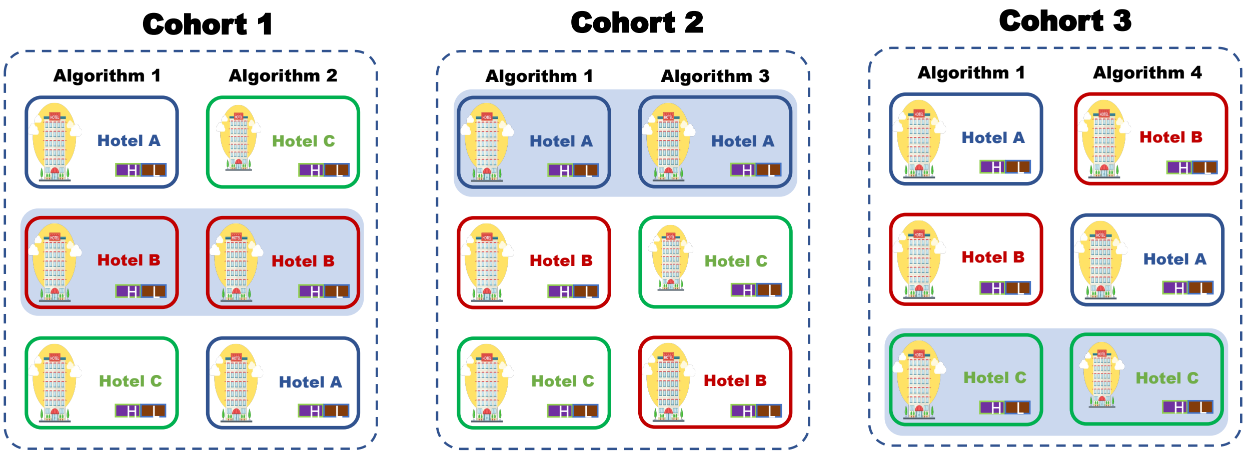


Figure 3: Panel variation in discrepancy between algorithm 1 and cohort rankings. In this example, for every hotel, there exists a cohort where algorithm 1 and cohort rankings are consistent.

We now present the intuition for our solution concept and then describe our approach to recover the true TATE based on data from a series of cohorts/experiments. Figure 3 builds on the example from §3.1 and illustrates how panel variation *across* cohorts can be leveraged to recover the true TATE. Recall that in our example, the demand for hotel B under algorithm 1 was unbiased because its ranking under algorithm 1 and its cohort ranking (ranking under the mix of algorithms 1 and 2) were the same. We can now extend this intuition to a panel of cohorts. That is, for each cohort, we can identify the items whose rankings in algorithm 1 are similar to cohort rankings and combine these item-specific demand functions to calculate the TATE for algorithm 1. In Figure 3, we have three cohorts, where the discrepancy between algorithm 1 rankings and cohort rankings for the three hotels varies across the cohorts. Notice that by pooling together the demand of hotel B in cohort 1, hotel A in cohort 2, and hotel C in cohort 3, we can compute the true TATE of algorithm 1 without any interference bias. In what follows, we formalize this idea.

Based on our demand model, we know that the state of item j, $\mathcal{S}_j$, is only a function of its own overall demand and does not depend on the state or the demand for other items on the platform. We now combine this implication with an additional assumption on the continuity of the demand function to propose a solution to infer the true TATE of an algorithm a based on data from a series of experiments in which a is not deployed fully. We first state the continuity assumption.

Assumption 2. *Probability of purchase conditional on consideration $\alpha(\mathbf{x}_j, \mathcal{S}_j)$ is a continuous function of $\mathbf{x}_j$ and $\mathcal{S}_j$.*

Given continuity, we can write our first result as:

Proposition 1. *Under Assumptions 1-2, if $\|P_j^a - P_j^{\boldsymbol{m}_c}\|_1 \to 0$ in cohort c, then*

$$q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) \to q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot)).$$

Proof. See Appendix A.1. □

The main implication of this result is that, if an item j's ranking within algorithm a and its *overall* ranking in a cohort c are similar (i.e., $P^a_j(\cdot) \approx P^{\mathbf{m}_c}_j(\cdot)$ or $\|P^a_j - P^{\mathbf{m}_c}_j\|_1 \to 0$), then the observed demand of this item under a in this cohort will be similar to its true demand when a is fully scaled up (when $m_a = 1$). Notice that continuity in the demand function allows us to extrapolate the demand from a setting where $P^{\mathbf{m}_c}_j(\cdot) \to P^a_j(\cdot)$ to the case where we fully scale up algorithm a. This results reflects the formalized version of the intuition we saw in §3.1, where the demand for hotel B was unbiased since its ranking under algorithm 1 and overall cohort ranking were similar.

While the example from §3.1 only focused on panel variation, in practice, we can also leverage cross-sectional variation within a cohort to correct for interference bias. Suppose an item j has high discrepancy in its algorithm and cohort ranking for some cohort c, i.e., $P^a_j(\cdot) \neq P^{\mathbf{m}_c}_j(\cdot)$. Even though the observed demand of j under a in cohort c is biased, we can still recover its true demand under a if there exists another item j' such that: (i) j' is perceived to be similar to j from the perspective of algorithm a, that is $P^a_{j'}(.) \approx P^a_j(.)$, and (ii) j and j' have similar features. i.e., $\mathbf{x}_j \approx \mathbf{x}_{j'}$. This is formalized in the following proposition.

Proposition 2. *Under Assumptions 1-2, if* $\|P^a_{j'} - P^{\mathbf{m}_c}_{j'}\|_1 \to 0$, $\|P^a_{j'} - P^a_j\|_1 \to 0$*, and* $\|\mathbf{x}_j - \mathbf{x}_{j'}\|_1 \to 0$ *then*

$$q^{(d)}_a(\mathbf{x}_{j'}; P^a_{j'}(\cdot), P^{\mathbf{m}_c}_{j'}(\cdot)) \overset{(1)}{\to} q^{(d)}_a(\mathbf{x}_{j'}; P^a_{j'}(\cdot), P^a_{j'}(\cdot)) \overset{(2)}{\to} q^{(d)}_a(\mathbf{x}_j; P^a_j(\cdot), P^a_j(\cdot)).$$

Proof. See Appendix A.2 □

Together, Propositions 1 and 2 suggest that we can use two types of variation to correct for interference bias at the item-level: (a) variation in the discrepancy between $P^a_j(\cdot)$ and $P^{\mathbf{m}_c}_j(\cdot)$ for similar types of items within a cohort c, and (b) variation in the discrepancy between $P^a_j(\cdot)$ and $P^{\mathbf{m}_c}_j(\cdot)$ for the same item across cohorts. Thus if we have data from multiple cohorts, then we can pool observations that satisfy the conditions described in Propositions 1 and 2 to recover the the algorithm-level TATE. We formalize this idea below.

Proposition 3. *Let J and C be the set of items and cohorts, respectively. We can recover the TATE of algorithm a if the number of cohorts $|C| \to \infty$ and $\forall\ \ \epsilon > 0, \forall\ j\ \in J$ either*

$$\exists\ c \in C \quad \textit{such that} \quad \|P^a_j - P^{\boldsymbol{m}_c}_j\| < \epsilon,$$

$$\textit{or},\ \exists\ j' \in J, c \in C \quad \text{such that} \quad \max\{\|\mathbf{x}_j - \mathbf{x}_{j'}\|, \|P^a_j - P^a_{j'}\|, \|P^a_{j'} - P^{\mathbf{m}_c}_{j'}\|\} < \epsilon.$$

Proof. See Appendix A.3 □

Proposition 3 shows that if for each item $j \in J$, the discrepancy between algorithm and cohort rankings is low in a subset of cohorts either for j or for another item j' that is similar to j, then we can recover the true TATE of algorithm a by aggregating the item-level demand estimates at the low-discrepancy observations. In principle, as the number of cohorts grows, our ability to recover TATE should improve. That said, in practice, the number of cohorts required to identify TATE reliably is an empirical question, and researchers/managers will need to examine whether their setting has sufficient panel and/or cross-sectional variation for this task. We discuss the empirical aspects of this issue in §7.3.

So far, we considered a case where there is only *indirect interference*. However, in many real-world applications, there could be direct interference. *Direct interference* is formally defined as a situation where the distribution of rankings for an item j under algorithm a (i.e., P_j^a) is a function of the mix of algorithms shown in cohort c (i.e., $\mathbf{m}_c$). As discussed earlier, this can happen if algorithm a uses one or more of the following variables as inputs into its ranking function: (1) aggregate performance metrics of an item j, e.g., total clicks, bookings (which are influenced by the other algorithms being concurrently tested, i.e., the cohort), or (2) cohort-specific state variables, $\mathcal{S}_j$, e.g., prices, room availability that vary as a function of $\mathbf{m}_c$. Our solution concept easily extends to settings where there is both direct and indirect interference. We refer interested readers to Appendix §A.4, where we present the generalized version of Proposition 3 and its proof under some additional regularity assumptions.

5 Empirical Setting and Data

We now apply the solution concept discussed above to data from a hotel booking website, hereafter referred to as "the platform". Users use the platform to find hotel stays for a given set of dates and destination. The data set includes all user interactions in the period between July 2014 and January 2015. The platform tracks users based on cookies and collects data regarding the query text (destination), submission date, the stay period, returned results, and users' interactions with the listings such as clicks and bookings. As the vast majority of queries are submitted through desktop browsers and since cookie tracking is more reliable on desktop devices compared to mobile devices, we focus on desktop devices in this paper.

During our observation period, the platform ran a series of randomized control trials to examine the performance of different ranking algorithms. As such, we have data on multiple cohorts. In §5.1, we explain the type of data collected by the platform. Then, in §5.2, we present a summary of the field experiments that ran on the platform.

5.1 Data

Typically, when users arrive at the platform, they query or search for a destination, stay period, and specify the number of guests. The website then returns a rank-ordered set of hotels. As is typical for such platforms, the website lists a picture of the hotel with some key characteristics such as the star rating and the price. The user can click on hotels returned in the query results and book the hotel through the platform. For each query made by a user, three types of data are recorded:

- Query-related data: When a user submits a query, all the query-related information such as the query text that typically refers to the destination, the timestamp of query submission, the requested stay period, and the number of guests are recorded in the data.
- Search results data: The rank-ordered list of hotels and hotel attributes.[9] This includes both time-invariant

[9] For each search request made by a user, the first page of results are recorded. The results for later pages are logged only if the user

Variable			*Across Days*		
	Mean	Standard deviation	Median	5th Percentile	95th Percentile
Queries	213468.55	87511.11	218884.50	73620.85	347132.05
Click/Queries	0.47	0.09	0.49	0.37	0.53
Bookings/Queries	0.02	0.01	0.02	0.01	0.03

Table 2: Summary statistics for queries, clicks, and bookings across days.

Variable			*Across Hotels*		
	Mean	Standard deviation	Median	5th Percentile	95th Percentile
Impressions	24167.74	62589.88	3169.00	64.00	110475.50
1000 x Clicks/Impr.	10.08	14.79	5.65	0.00	35.34
1000 x Bookings/Impr.	0.34	1.49	0.00	0.00	1.63

Table 3: Summary statistics for the average number of impressions, click, and booking rate across hotels.

features of the hotel such as location and time-varying variables such as price and room availability.

- Clicks and booking data: The data record all the interactions of the user with the query results, including clicks on hotel listings that enable the user to view hotel details, the number of hotel listings viewed, and any bookings.

figures/aggregate/queryTextRankECDF-eps-converted-to.pdf

Figure 4: The empirical CDF of query texts in our data. The x-axis is in log scale. The top 18 query texts are responsible for 25% of the total queries. The dashed lines represent the first, second, and third quantiles.

Overall, we have data on 43 million queries from this period, which led to more than 20 million clicks and 900 thousand bookings. Table 2 presents summary statistics of the number of queries, clicks, and bookings at the daily level, over our observation period. On average, each day in our observation period received

navigates to them. In §7, we explain how this is taken into account in our discrepancy measures.

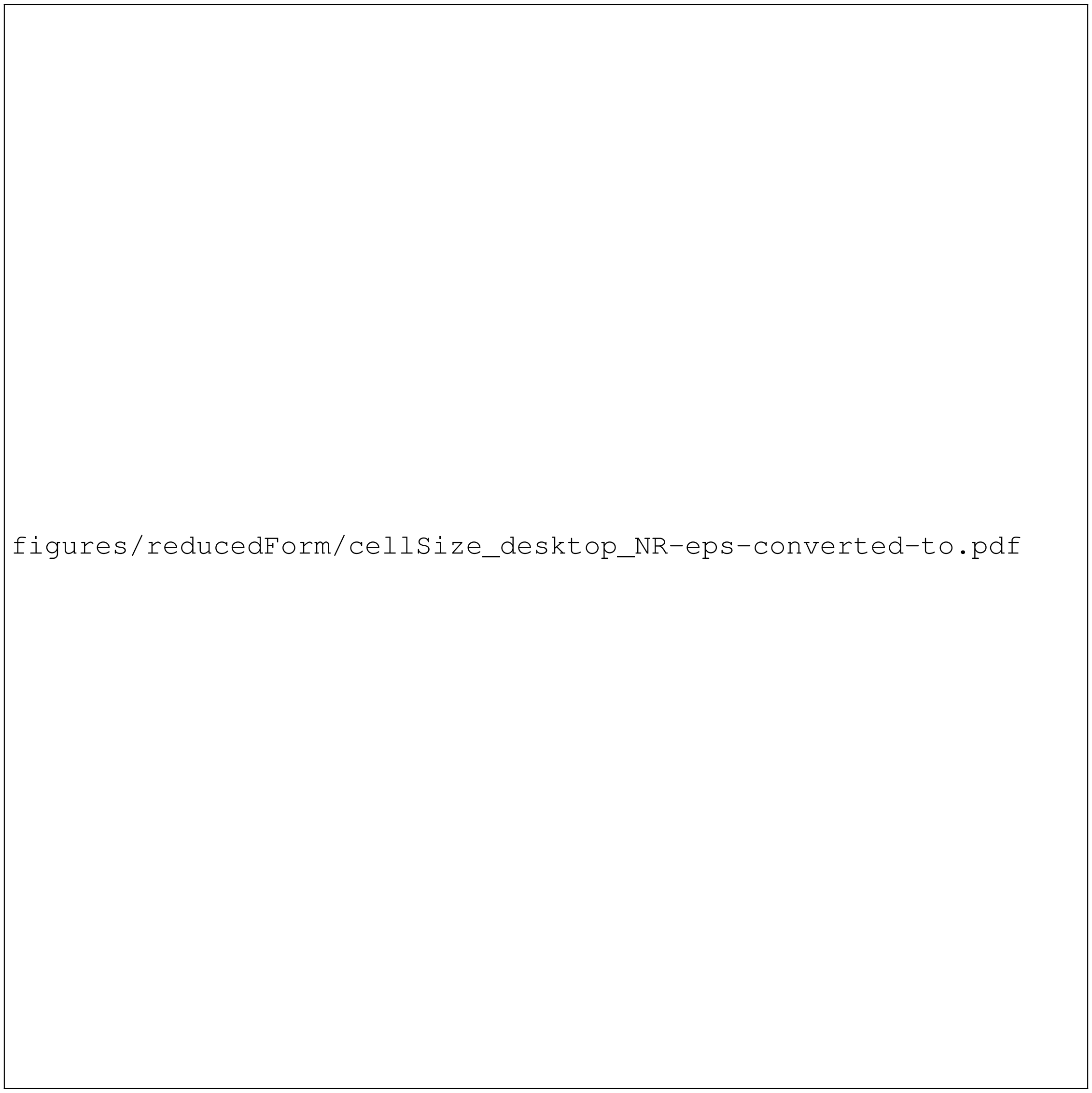
over 200,000 queries, and each query received 0.47 clicks and 0.02 bookings. Next, Table 3 summarizes hotel-level data. A hotel appeared, on average, in 24,167.74 queries though we see significant variation across hotels. The hotel-level ratios of clicks/impressions and bookings/impressions are both low, reflecting that users click only on a relatively small number of hotels listed in response to any search query. Finally, as Figure 4 illustrates, queries are unevenly distributed across destinations, which is one factor that drives the variation in the number of search results in which a hotel appears.

5.2 Field Experiments

During the observation period between July 2014 to January 2015, the platform ran a series of experiments to assess the performance of a large set of ranking algorithms. As described in Definition 1, each experiment denotes a cohort, where users are randomly assigned to one of the ranking algorithms with a fixed probability that reflects the experiment cell sizes. There are a total of 16 cohorts in the observation period, and the length of these cohorts vary from a couple of days to a couple of weeks. In each cohort, the platform ran between two and four algorithms concurrently. The proportions of traffic assigned to each algorithm varied across cohorts but were held constant within a cohort. Data from three cohorts (cohorts 9, 13, and 14) were not collected because the experimentation platform failed. In 13 of the remaining 14 cohorts, the platform tested one of their core ranking algorithms, “Genesis”, against a wide mix of other ranking algorithms.[10]

Figures 5a – 5b depict the portion of queries assigned to the Genesis algorithm and the click-through rate of users in the Genesis algorithm in our data. We see that there is a lot of variation both in terms of the experiment cell size and the performance of the Genesis algorithm across cohorts. Notice that there are some dramatic changes in the performance between the last day of one cohort and the first day of the subsequent cohort. This is particularly prominent for the last two cohorts, where the performance and allocation of users to the Genesis algorithm are drastically different across cohorts (the click-through rate under the Genesis drops by 11% during the one day between the end of cohort 15 and the beginning of cohort 16). The fact that a change in the portion of queries assigned to the Genesis algorithm went along with a change in the click-through rate of this algorithm suggests the possibility of interference bias. We examine this possibility in the next section.

[10] Notice that the solution approach presented in §4.3 is agnostic to the number, cell-size, and exact nature of the different algorithms in any given cohort. So we do not present any additional details on the details or allocation of the different algorithms tested in the data.


figures/reducedForm/cellSize_desktop_NR-eps-converted-to.pdf

(a) The portion of queries that are assigned to the Genesis algorithm.

figures/reducedForm/clickRate_desktop_NR-eps-converted-to.pdf

6 Reduced-form Evidence for Interference Bias

In this study, our goal is to infer the TATE of Genesis from the available data. Notice that this would be trivial if – (1) there was at least one cohort where Genesis was the only algorithm being tested, or (2) if there were no interference bias. We know that condition (1) is not true; in no cohort do we see Genesis as the sole algorithm being tested. Therefore, we examine if condition (2) is true or not, i.e., we analyze the data for the presence of interference bias. If we find strong evidence for interference bias, then the observed click-through and booking rates (i.e., the observed TATEs) of Genesis in any given cohort are likely to be biased, and we need to apply the solution proposed in §4.3 to recover the unbiased TATE of Genesis. Therefore, we now test for and present evidence for the presence of interference bias in our data. First, in §6.1 we present evidence from user-level click-through and booking rates. Next, in §6.2, we present evidence based on hotel-level click-through and booking rates.

6.1 Evidence from Click-Through Rate and Booking Rate at User Level

In this analysis, we focus on the last two cohorts, cohorts 15 and 16. These two cohorts are not only consecutive but also the two longest cohorts in our data. Further, there is a significant change in the portion of queries assigned to Genesis across these two cohorts. As such, they present an ideal opportunity to examine the presence (if any) of interference bias.

Notice that both cohorts 15 and 16 featured the Genesis algorithm and two others, which we refer to here as algorithms 1 and 2. Figures 6a–6b present the portion of users assigned to these three algorithms and the respective click-through rates during the last two cohorts.[11] The abrupt change in performance of the Genesis algorithm across the last two cohorts is notable in Figure 6b. As discussed in §4, such discontinuities can occur if the discrepancy between the ranking of a hotel in Genesis and its average ranking in the cohort varies across the two cohorts and are indicative of the presence of interference bias. However, we first need to ensure that this discontinuity in the performance of the Genesis algorithm is not a result of platform or demand-side changes that could have occurred at the same time as the cohort change. To examine this alternative hypothesis, we consider the performance of the two other algorithms that were also tested during the last two cohorts. If the discontinuous change in the performance was due to a change in other platform-related characteristics or due to a change in demand, such changes should also be reflected in the performance of algorithms 1 and 2. However, as Figure 6 shows, this is not the case.

To validate the robustness of this finding, we also estimate the following difference-in-differences model using data from the last two cohorts:

$$\log(Y_{it}) = \alpha \cdot \mathbb{1}_{\{t \geq t^*\}} + \beta \cdot \mathbb{1}_{\{t \geq t^*\}} \cdot \mathbb{1}_{\{a_i = Genesis\}} + \eta_{a_i} + \eta_t, \tag{7}$$

where i, t, and a_i index users, days, and the algorithm that user i is assigned to. Y_{it} is an outcome of interest, i.e., click-through or booking rate by user i in day t and is defined as $\frac{Total\ Clicks+1}{Total\ Queries+1}$ or $\frac{Total\ Bookings+1}{Total\ Queries+1}$,

[11] A third algorithm was also tested in the second-to-last cohort. However, that algorithm was not present in the last cohort. Thus, it is not relevant to our analysis here.

figures/reducedForm/cell_size_desktop_last_two-eps-converted-to.pdf

(a) The portion of queries assigned to different algorithms.

figures/reducedForm/clickRate_desktop_last_two-eps-converted-to.pdf

	Dependent variable:			
	log(User click-through rate)		log(User booking rate)	
	(1)	(2)	(3)	(4)
Last Cohort x Genesis (β)	-0.119^{***}	-0.107^{***}	-0.041^{***}	-0.042^{***}
	(0.003)	(0.003)	(0.003)	(0.003)
Last Cohort (α)	0.002		0.012^{***}	
	(0.003)		(0.002)	
Algorithm FE	X	X	X	X
Day FE		X		X
Observations	1,688,617	1,688,617	1,688,617	1,688,617
R^2	0.006	0.025	0.001	0.002
Adjusted R^2	0.006	0.025	0.001	0.002
Residual Std. Error	0.797 (df = 1688612)	0.789 (df = 1688546)	0.693 (df = 1688612)	0.692 (df = 1688546)
Note:				*p<0.1; **p<0.05; ***p<0.01

Table 4: Measuring the differential change in the click-through and booking rate using a difference-in-differences design in the last two cohorts across different algorithms.

respectively. Also t^* is the first day of the last cohort, and $\mathbb{1}_{\{t \geq t^*\}}$ and $\mathbb{1}_{\{a_i = Genesis\}}$ are dummy variables that are equal to 1 if the observation belongs to the second cohort and if the user was assigned to the Genesis algorithm, respectively. Finally, η_{a_i} and η_t are algorithm and day fixed effects. The estimates from the model shown in Equation (7) are reported in Table 4. The first column of results indicate that the click-through rate increases slightly as we move to the last cohort by $\exp(0.002) - 1 = 0.2\%$, however, the click-through rate of those users assigned to the Genesis algorithm drops by $\exp(-0.119) - 1 = -11.2\%$. When we control for day fixed-effects in the second column of Table 4, the results are similar. Similar patterns also hold for bookings when we move to columns (3)-(4) of Table 4.

6.2 Evidence from Click-Through Rate and Booking Rate at Hotel Level

If the patterns observed in Figure 6 and Table 4 were due to interference bias in that the discrepancy between hotels' rankings in the Genesis algorithm and the average ranking in the cohort changed across the two cohorts, then such discontinuous shifts in performance should not occur for hotels that had low discrepancy between the ranking in Genesis and their overall cohort rankings in the two cohorts.

To test this hypothesis, we first need to define an empirical measure of the discrepancy between the distribution of rankings generated by Genesis and the cohort rankings for a given hotel. More precisely, we need a distance metric between $P_j^{a_c}(\cdot)$ and $P_j^{\mathbf{m}_c}(\cdot)$ for each hotel j in cohort c. We consider the following simple discrepancy metric:

$$d(P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot)) = \left| \log \left(\frac{\mathbb{E}_{P_j^{a_c}}[r_j]}{\mathbb{E}_{P_j^{\mathbf{m}_c}}[r_j]} \right) \right| = \left| \log \left(\frac{\sum_{r=1}^{\infty} r \cdot P_j^{a_c}(r)}{\sum_{r=1}^{\infty} r \cdot P_j^{\mathbf{m}_c}(r)} \right) \right|, \tag{8}$$

which is simply the absolute value of the $\log$ ratio of the expected ranking of a hotel j by algorithm a divided by the expected ranking of the same hotel in the entire cohort. If $P_j^{a_c}(\cdot) = P_j^{\mathbf{m}_c}(\cdot)$, that is there is no discrepancy, then $d(P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot)) = 0$. This metric captures the discrepancy of the average ranking of a hotel j in the overall cohort and under the Genesis algorithm. For instance, if $\log\left(\frac{\sum_{r=1}^{\infty} r \cdot P_j^{a_c}(r)}{\sum_{r=1}^{\infty} r \cdot P_j^{\mathbf{m}_c}(r)}\right) = 0.2$, then $\frac{\sum_{r=1}^{\infty} r \cdot P_j^{a_c}(r)}{\sum_{r=1}^{\infty} r \cdot P_j^{\mathbf{m}_c}(r)} = \exp(0.2) = 1.22$, which means that the hotel's average rank under the Genesis algorithm is 22% higher than its average rank in the overall cohort. In general, for small values of discrepancy, $d(P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot))$, the average rank under Genesis and the average overall rank deviate by about $d(P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot)) \times 100$ percent. While this metric focuses on the discrepancy between average rankings, later in §7, we include a richer set of discrepancy metrics that capture the full range of differences between the two ranking distributions beyond just the mean/average ranking.

Figure 7 presents the distribution of the discrepancy metric in Equation (8) for the Genesis algorithm across the last two cohorts. As expected the discrepancy shrinks in the last cohort because the Genesis algorithm was scaled up (see Figure 6a for experiment cell sizes).

	Dependent variable:		
	Log(Hotel click-through rate)		
	(1)	(2)	(3)
Last cohort x High overall rank discrepancy		-0.508^{***}	-0.508^{***}
		(0.013)	(0.013)
Last cohort	-0.291^{***}	-0.070^{***}	-0.070^{***}
	(0.006)	(0.006)	(0.006)
High overall rank discrepancy		0.591^{***}	
		(0.011)	
Constant	-4.254^{***}	-4.511^{***}	
	(0.005)	(0.007)	
Hotel FE			X
Observations	85,292	85,292	85,292
R^2	0.017	0.052	0.701
Adjusted R^2	0.017	0.052	0.402
Residual Std. Error	1.106 (df = 85290)	1.086 (df = 85288)	0.862 (df = 42644)
Note:			*p<0.1; **p<0.05; ***p<0.01

Table 5: Change in click-through rate of hotels with low/high overall ranking discrepancy across the last two cohorts.

If the patterns in Figure 6b arise because of the discrepancy between hotels' ranking for Genesis relative to the entire cohort, we should observe a discontinuity in click-through rate only for hotels that had a high discrepancy in at least one of the two cohorts. To that end, we consider the following metric to capture the overall discrepancy between Genesis rankings and cohort rankings across the last two cohorts (cohorts 15 and 16):

$$\tilde{d}_j = \max\{d(P_j^{a_{15}}(\cdot), P_j^{\mathbf{m}_{15}}(\cdot)), d(P_j^{a_{16}}(\cdot), P_j^{\mathbf{m}_{16}})(\cdot)\}, \tag{9}$$

which calculates the maximum discrepancy between the algorithm a (Genesis in this case) and the cohort

figures/reducedForm/discrepancy_last_two_cohort_NR_user-eps-converted-to.pdf

Figure 7: Discrepancy between Genesis and cohort hotel rankings across last two cohorts. As expected the discrepancy between Genesis and cohort rankings shrinks in the last cohort (cohort 16) because the Genesis algorithm was scaled up in the last cohort, see Figure 6a.

rankings across the two cohorts (cohorts 15 and 16) for a given hotel j. The distribution of $\tilde{d}_j$, hereinafter the overall discrepancy.

We now examine if the overall ranking discrepancy across hotels explains the change in CTR. We partition the hotels into two sets, $\mathcal{H}_l$ and $\mathcal{H}_h$, based on a median split on the overall discrepancy $\tilde{d}_j$. The hotels below the median ($\mathcal{H}_l$) had low ranking discrepancy in both cohorts and those above the median ($\mathcal{H}_h$) had high ranking discrepancy in at least one of the two cohorts. Based on the discussion above, we would expect the discontinuity in Figure 6b to stem from hotels that had high overall ranking discrepancy. To verify this, we examine the click-through rate of hotels with low vs. high overall ranking discrepancy across the last two

figures/reducedForm/overall_disc_click_rate_user-eps-converted-to.pdf

Figure 8: Change in click-through rate for hotels with low/high overall ranking discrepancy across the last two cohorts, under Genesis.

cohorts, for Genesis, in Figure 8.[12] As we can see, the discontinuity in click-through rate only occurs for hotels with high overall ranking discrepancy.

We now examine if the overall ranking discrepancy across hotels explains the change in CTR. We partition the hotels into two sets, $\mathcal{H}_l$ and $\mathcal{H}_h$, based on a median split on the overall discrepancy $\tilde{d}_j$. The hotels below the median ($\mathcal{H}_l$) had low ranking discrepancy in both cohorts, and those above the median ($\mathcal{H}_h$) had high ranking discrepancy in at least one of the two cohorts. Based on the discussion above, we would expect the discontinuity in Figure 6b to stem from hotels that had a high overall ranking discrepancy. To verify this, we examine the click-through rate of hotels with low vs. high overall ranking discrepancy across the last two cohorts, for Genesis, in Figure 8.[13] As we can see, the discontinuity in click-through rate only occurs for hotels with high overall ranking discrepancy.

[12] The click-through rate for a hotel is the number of clicks on that hotel divided by the total number of times the hotel was viewed on a page listing.

[13] The click-through rate for a hotel is the number of clicks on that hotel divided by the total number of times the hotel was viewed on a page listing.

To verify the robustness of the patterns in Figure 8, we need to control for hotel fixed effects. Therefore, we estimate the following difference-in-differences model to measure the change in the click-through rate of hotels using only those query results generated by the Genesis algorithm:

$$\log(Y_{jc}) = \alpha \cdot \mathbb{1}_{\{c=16\}} + \beta \cdot \mathbb{1}_{\{c=16\}} \cdot \mathbb{1}_{\{j \in \mathcal{H}_h\}} + \eta_j + \epsilon_{jc}, \tag{10}$$

where j refers to hotels and $c = 15, 16$ indexes cohorts. Also, $Y_{jc} = \frac{\text{1+clicks on hotel } j \text{ in cohort } c}{\text{1+total impressions of hotel } j \text{ in cohort } c}$ is the click-through rate of hotel j in cohort c by users assigned to the Genesis algorithm.[14] Here, η_j denotes the hotel fixed effect. We estimate Equation (10) using all hotels that appeared in at least 100 query responses (i.e., generated at least 100 impressions), across both cohorts and report the results in Table 5. Column (1) of Table 5 shows that the hotel click-through rate drops on average by $\exp(\text{-}0.291) - 1 = \text{-}25.25\%$ as we move from cohort 15 to cohort 16, an average that is calculated across hotels. Column (2) shows that this drop in click-through rate is mainly explained by the change in click-through rates of hotels that had high overall rank discrepancy ($\exp(\text{-}0.508) - 1 = \text{-}39.83\%$) compared to a much lower common trend ($\exp(\text{-}0.070) - 1 = \text{-}6.7\%$). In Column (3) we allow for hotel fixed effects and the estimates remain identical to that of Column (2). These results suggest that the change in performance across cohorts is more pronounced for hotels for which the overall discrepancy was high, and that conditional on the overall discrepancy being low, a hotel's click-through rate seems to be fairly stable.[15]

In summary, our results in this section highlight that: (1) there is heterogeneity in the changes in click-through rate across hotels, (2) the change in a hotel's click-through rate is systematically correlated with its overall discrepancy across the two cohorts, and (3) the magnitude of change is smaller for hotels that have lower overall discrepancy, i.e., a hotel's performance tends to remain stable when its overall discrepancy is low. Together, these findings confirm the presence of interference bias in the data and suggest that the discrepancy in cohort and algorithm rankings is the main source of this interference bias.

7 TATE Prediction Algorithm

Building on the findings above, we now develop a machine learning model to predict the TATE of an algorithm when it is scaled to 100% (that is when the treatment is applied to the full population) based on data from a series of cohorts where the algorithm was served to different subsets of the population. In this section, we focus on clicks as the outcome variable of interest. Later, in §8.3, we explain how we expand the method to include bookings and show the results for booking.

We define the set of inputs that go into our algorithm and relate them to the variables discussed in §4 and §6. First, recall that $\mathbf{x}_j$ is the set of item/hotel attributes. Second, let $\mathbf{g}_t$ be a set of seasonality variables that affect users' demand for hotels on a given day t. This allows us to capture changes in users' demand across destinations and hotels over time. We discuss these measures in detail in §7.2 and Appendix C. Finally, let $\mathbf{d}^a_{jc_t}$ be a set of discrepancy measures that capture the deviation of algorithm a's rankings from cohort c_t's

[14] We considered hotels that had at least 100 impressions in each of the cohorts.

[15] Similar patterns are shown for booking rates in Appendix B.

rankings on day t and is calculated at the hotel-cohort level. Notice that c_t indexes the cohort at time t. Now, we can write the expected demand or total clicks generated under algorithm a on a given day as:

$$\mathbb{E}[\text{Clicks}_t^a] = Q_t^a \cdot \sum_{j=1}^{J} \left[\mathbb{P}[\text{Imp}|\text{Query}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}_{jc_t}^a = \mathbf{0}] \cdot \mathbb{P}[\text{Click}|\text{Imp}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}_{jc_t}^a = \mathbf{0}] \right], \tag{11}$$

where:

- Q_t^a is the total number of queries assigned to algorithm a on day t. Of course, when the algorithm is scaled up to the entire platform, all the queries are assigned to this algorithm, i.e., $Q_t^a = Q_t$.
- $\mathbb{P}[\text{Imp}|\text{Query}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}_{jc_t}^a = \mathbf{0}]$ is the probability that an impression for hotel (or item) j appears in the query results under algorithm a, conditional on item features $\mathbf{x}_j$ and daily seasonality variables $\mathbf{g}_t$, when the discrepancy between the algorithm and the cohort is zero (i.e., $\mathbf{d}_{jc_t}^a = \mathbf{0}$).
- $\mathbb{P}[\text{Click}|\text{Imp}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}_{jc_t}^a = \mathbf{0}]$ is the probability of click on an impression for hotel (or item) j with features $\mathbf{x}_j$, seasonality variables g_t, when the impressions positions follow the equilibrium distribution of rankings for j under algorithm a. As before, this is achieved by considering the case where the discrepancy between the algorithm and the cohort is zero (i.e., $\mathbf{d}_{jc_t}^a = \mathbf{0}$).

The term inside the summation is the expected demand of hotel j under algorithm a, when a is scaled up to 100%.

Note that the total number of queries that the platform receives on a given day (Q) is assumed to be exogenous. The number of people searching for accommodation in a given location on a particular day is not a function of the algorithm used by the platform, which seems reasonable (at least in the short run). However, the number of impressions that a hotel receives is endogenous, i.e., it is a function of the algorithm's ranking function. As such, this is an equilibrium outcome and needs to be inferred from the data.[16] Similarly, the number of clicks that a hotel receives is a function of both the distribution of rankings $P_j^a(\cdot)$ as well as the state S_j of the hotel, both of which are equilibrium outcomes. Therefore, to calculate $\mathbb{E}[\text{Clicks}_t^a]$, we need consistent estimates of $[\mathbb{E}[\text{Imp}|\text{Query}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}_{jc_t}^a = \mathbf{0}]$ and $\mathbb{E}[\text{Clicks}|\text{Imp}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}_{jc_t}^a = \mathbf{0}]$. We use two separate deep neural networks for these two estimation tasks. The first model predicts the share of impressions for each hotel under a given algorithm. The second model predicts the click-through rate for each hotel for a given number of impressions.

Finally, for training and validation, we use the data from Genesis across all cohorts, except the last cohort, which is used as a hold-out sample to demonstrate the performance of the algorithm. Figure 9 highlights the portion of the data used for training and validation and the hold-out sample.

The rest of this section is organized as follows. In §7.1, we present the details of the objective function and architecture of the neural networks employed for these two estimation tasks. Next, in §7.2 we describe

[16] Notice that an item's distribution of positions, $P_j^a(\cdot)$, and its state S_{jt} (which can include variables such as prices or the distributions of rooms available) are equilibrium outcomes and should not be fed to the model directly as they may change with seasonality or as the mix of algorithms used. In principle, it is possible to train a separate model and learn these variables as a function of the exogenous variables $\mathbf{x}_j$, $\mathbf{g}_t$ and then feed those predictions as features into our models of impressions and clicks. However, doing so is redundant since there is no new information added from this step because $\mathbf{x}_j$ and $\mathbf{g}_t$ are directly used as inputs into our models.

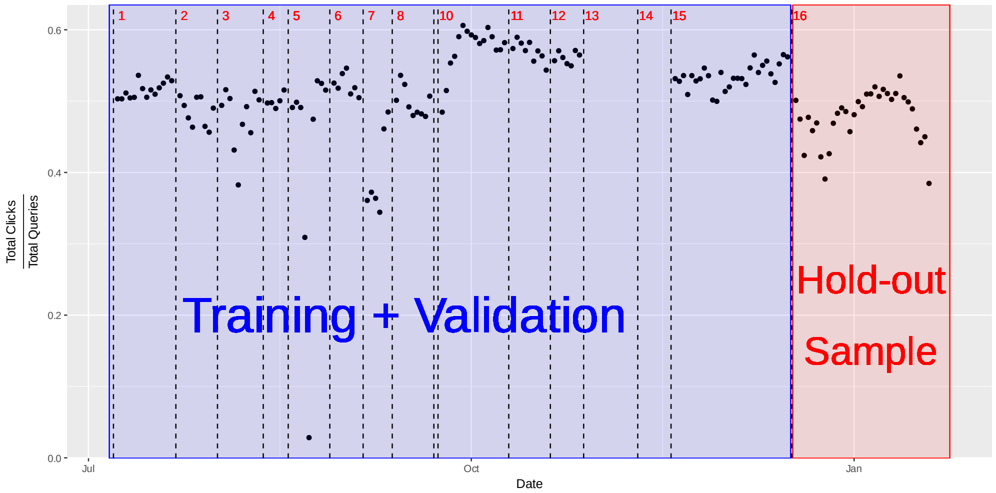


Figure 9: We use all cohort data except for the last cohort for training and validation. The last cohort is used as a hold-out sample to demonstrate the external validity of our model.

the hotel, seasonality, and discrepancy features. In §7.3, we present the sources of variation in the data that helps us recover TATE.

7.1 Objective Functions and Network Architecture

We first discuss the objective function for the estimation task of $\mathbb{E}[\text{Imp}|\text{Query}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t} = \mathbf{0}]$. Let $p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$ be the probability that a hotel j with features $\mathbf{x}_j$ appears in the result of a query on a day characterized by the seasonality vector $\mathbf{g}_t$, with a set of algorithm-cohort discrepancy features $\mathbf{d}^a_{jc_t}$. We parameterize $p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$ as a neural network. Further, let k^a_{jt} denote the number of impressions generated through the Genesis algorithm for a hotel j on day t, when Q^a_t queries were assigned to Genesis. Modeling this process as a binomial random variable, the log-likelihood of observing k_{jt} impressions for a hotel j, out of the Q^a_t queries is given by:

$$\mathcal{L}(Q^a_t, k^a_{jt}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t}; p_0) = \log\left[\binom{Q^a_t}{k^a_{jt}} p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})^{k^a_{jt}} \left(1 - p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})\right)^{Q^a_t - k^a_{jt}}\right]. \tag{12}$$

The main advantage of this objective function (over other commonly used ones, e.g., lognormal) is that it accommodates both zero and non-zero outcomes. Our objective is to maximize the sum of the log-likelihood:

$$\mathcal{L} = \sum_{t=1}^{T} \sum_{j=1}^{J} \mathcal{L}(Q^a_t, k^a_{jt}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t}; p_0), \tag{13}$$

where we sum over hotels and all the days in our training data. Further, note that each observation is at the hotel-day (jt) level.

Recall that $p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$ is parameterized as a neural network. Next, we tune this neural network by maximizing the log-likelihood shown in Equation (13) on our training data. A schematic view of the neural network architecture is presented in Figure 10. For each hotel-day observation under the Genesis algorithm, the neural network outputs the probability of being shown (i.e., impression) in response to a query, $p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$, and tunes it to maximize the log-likelihood for the data set.

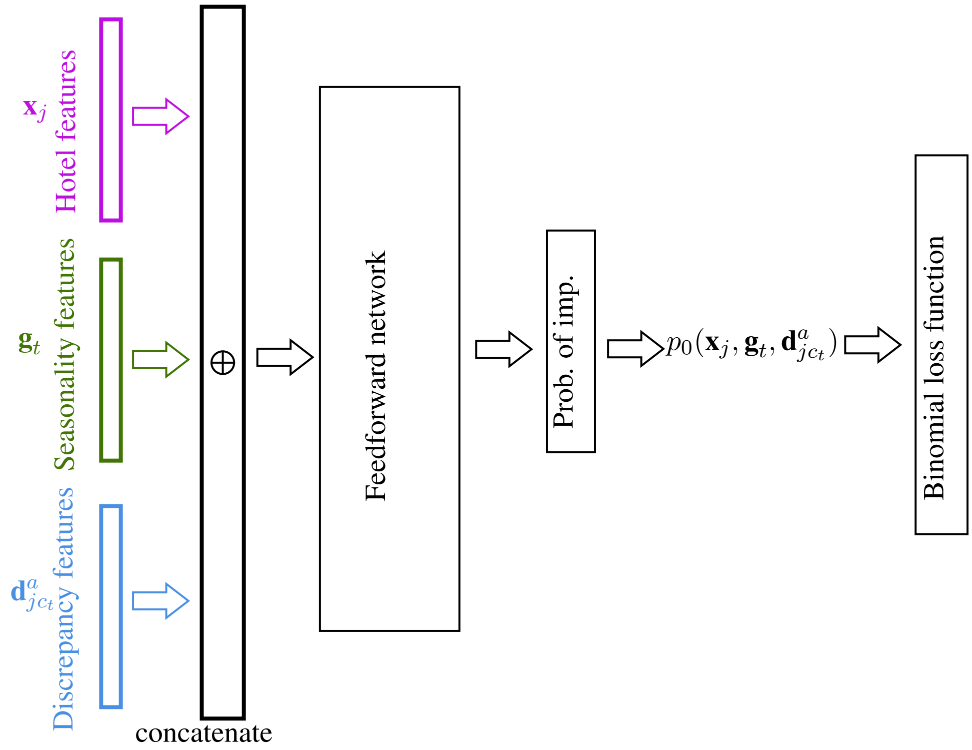


Figure 10: A schematic view of the neural network structured used for learning the impression probability.

We also train another neural network to learn $\mathbb{E}[\text{Click}|\text{Imp}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t} = \mathbf{0}]$. For this task, we use the same network architecture and feature set and an objective function analogous to the one used for the impression-rate prediction task described above. Briefly, let $p_1(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$ be the click probability on an impression under the Genesis algorithm for hotel j represented by features $\mathbf{x}_j$ on a day characterized by the seasonality vector $\mathbf{g}_t$, with a set of algorithm-cohort discrepancy features $\mathbf{d}^a_{jc_t}$. We then specify the log-likelihood of observing a given number of clicks using a similar likelihood structure as in Equation (12) and then maximize the sum of this log-likelihood for our data set. Finally, note that the hyper-parameters for both networks, such as the learning rate, drop out rates, number of units in each layer, and the number of layers, are tuned using five-fold cross validation.[17]

[17] We perform cross-validation by block sampling observations at the hotel-cohort level rather than hotel-day level. This is due to two key reasons: (a) the true variation in discrepancy for each hotel occurs at the cohort rather than day level, and (b) the goal is to extrapolate across cohorts.

7.2 Feature Sets

We now provide a brief overview of the three sets of features that serve as inputs to both the neural network models described above (and refer readers to Appendix C for details).

- Hotel features ($\mathbf{x}_j$): We compile a set of both observable and latent features of hotels, which together capture the similarity between a given pair of hotels. For the set of observable features, we include variables based on the observed outcomes related to the hotel in the training data. These include: (1) the average click-through rate of the hotel, (2) the impression share of the hotel, (3) the average ranking discrepancy under the Genesis algorithm across all cohorts, (4) the average rank of the hotel under the Genesis algorithm, (5) the average cohort rank across all cohorts for each hotel, i.e., $\bar{r}_j = \frac{1}{|\mathcal{C}|}\sum_{c\in\mathcal{C}}\sum_{r=1}^{\infty} rP_j^{\mathbf{m}_c}(r)$. All these variables are calculated at the hotel level for the entire training data.[18] For the latent hotel features, we learn two representation vectors for each hotel using the GloVe method (Pennington et al., 2014). GloVe recovers low-dimensional representations of objects based on co-occurrence data. It was initially applied to natural language processing and is now used extensively in other settings in marketing, e.g., brand perception (Dzyabura and Peres, 2019), online reviews (Chakraborty et al., 2021), and dynamic user interests (Dhillon and Aral, 2021). However, the general idea of using co-occurrence matrices and multi-dimensional scaling to obtain a low-dimensional representation of products and understanding market structure has existed long before; see (Netzer et al., 2012) for an example.

  The first representation vector captures the similarity between a given pair of hotels j and j' from the perspective of a given algorithm (Genesis in our case) based on how often the two hotels appear on the same results page. The intuition is that if two hotels often co-occur in the results of the Genesis algorithm, then these two hotels are perceived to be similar by the algorithm. The second representation vector is learnt by applying the same idea to clicks. It captures the similarity of two hotels based on the extent to which users perceive the two hotels to be similar, i.e., the extent to which the two hotels are clicked together by a given user (co-occurrence of clicks). See Appendix §C.1 for details.[19]

- Seasonality features ($\mathbf{g}_t$): The performance of the algorithm can shift due to the seasonality or due to a change in the type of users that arrive at the platform. Therefore, we need to capture this heterogeneity in seasonality. A naive way to do this is to use day or week dummies. However, that would only allow us to predict TATE for days in the training data but not for days in the test data. Therefore, we develop a representation vector for each day in our data based on the distribution of queries for the top 5000

[18] We do not use features such as hotel amenities, user ratings, latitude/longitude, etc., because these variables are not available for a large fraction of the hotels in our data. Therefore we have to rely on observed outcomes in the data to generate both observable and latent hotel features.

[19] If the rankings are endogenous, i.e., there is direct interference between algorithms, then using only a single cohort for calculating the co-occurrence matrix could bias our estimates of $\mathbf{x}_j$ and adversely affect the algorithm's performance. Hence, to mitigate this, we calculate these features using all cohorts in the training data. In the case where rankings are not endogenous, and interference can only come from indirect sources (i.e., through the state variables), using data from just one cohort is not an issue.

destinations.[20] This method requires only mild assumptions on the exogeneity of latent demand (i.e., the distribution of queries on a given day is not a function of the algorithms used by the travel website on that day, which is reasonable). Please see Appendix §C.2 for further details.

- Ranking discrepancy metrics ($\mathbf{d}^a_{jc_t}$): For each hotel, on a given day t in cohort c_t, we use a vector of discrepancy metrics to measure the ranking discrepancy between Genesis and cohort rankings. In §6.2, we used a simple discrepancy metric that compared the average ranking in Genesis vs. the average cohort ranking for each hotel. We now augment this with additional metrics that better capture the distance between the distribution of rankings in the cohort and Genesis rankings. Specifically, we now include the log ratio of the corresponding deciles across Genesis rankings and cohort rankings and the log ratio of the share of impressions across Genesis and cohort (in addition to the log ratio of expected rankings described previously). All these features are described in detail in Appendix §C.3.

7.3 Identification

Our TATE prediction algorithm exploits two types of variation to estimate the true TATE: (1) the variation in the rank discrepancy of the same hotel across different cohorts and (2) the variation in rank discrepancy for similar types of hotels (similar **x**s). We now discuss these two sources of variation in detail and document the extent of variation in these measures in our data.

First, we use all cohorts in the training data, and for each hotel-cohort combination, we compute the discrepancy measure defined in Equation (8) between the Genesis algorithm and the overall cohort ranking. Then, for each hotel, we calculate the mean, maximum, and minimum discrepancy between the Genesis and the overall cohort rankings across all available cohorts and present it in Figure 11. On average, for any given hotel, there exists a reasonable amount of discrepancy between the overall rankings and cohort rankings (as shown in panel 1). Similarly, the maximum discrepancy across cohorts can be quite high (panel 2). However, from an identification perspective, the main question of interest is – does there exist a cohort c such that $\mathbf{g}_t, \mathbf{d}^a_{jc_t}) \approx 0$ for all $j \in J$? The rightmost panel of Figure 11 demonstrates that this condition is largely satisfied in our data. The minimum discrepancy across cohorts is close to zero for almost all hotels. Thus, for most hotels in our data, there exists a cohort where the cohort rankings and Genesis rankings are similar. Such observations are important for determining the performance of the Genesis algorithm when it is scaled up.[21] Intuitively, this can be interpreted as exploiting the time-series variation within hotels in the TATE prediction task.

Another useful source of variation is the difference in ranking discrepancy for similar hotels within the same cohort. Consider two hotels j and j' that are ranked similarly by the Genesis algorithm and have

[20] In principle, $\mathbf{g}_t$ can also include other query-related information such as the number of travelers, number of days of stay, the lead time from booking to stay. However, including these additional variables did not significantly change the performance of our machine learning model. So we skip them for simplicity.

[21] In §6.2, we saw that some hotels had low discrepancy in the Genesis and cohort ranking in the last two cohorts, which allows us to infer the treatment effect of the Genesis algorithm for those hotels. However, for hotels that had high overall discrepancy in the last two cohorts, we need to ensure that there exist cohorts where these hotels have low discrepancy in order to measure the TATE of the Genesis algorithm.

figures/reducedForm/variation_discrepancy_across_cohorts_user-eps-converted

Figure 11: Variation in the discrepancy, $d(P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot))$, between the Genesis algorithm (denoted as a) and the overall cohort rankings for hotels across all cohorts in the training data, where Genesis was tested. The left panel shows that there is a lot of variation in rank discrepancy across hotels on average. The middle panel shows that there are periods in which the discrepancy for each hotel could be very high, and these observations could lead to biased estimates of Genesis's performance. The right panel shows that there exist cohorts where the rank discrepancy between the Genesis and cohort rankings is very small for almost all hotels. These observations are useful for estimating the true performance of the Genesis algorithm when it is scaled up.

similar features. Further, suppose that j's ranking under Genesis and a given cohort c are similar, i.e., $d(P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot)) \approx 0$, while j''s ranking across Genesis and the cohort is high, i.e, $d(P_j^{a_c}(\cdot), P_{j'}^{\mathbf{m}_c}(\cdot)) > 0$. In this case, because hotels j and j' are similar, we can use the performance of j at low discrepancy values to predict the performance of j'. Intuitively, this is similar in spirit to exploiting the cross-sectional variation across similar hotels within a cohort.

To illustrate this variation, for each hotel, we find its K-nearest neighbors in the hotel feature space ($\mathbf{x}_j$) and calculate the minimum ranking discrepancy across that hotel and its K-nearest neighbors within each cohort in the training data. We refer to this measure as the "K-discrepancy score" for each hotel. A low K-discrepancy score for a hotel in a given cohort means that either the hotel itself or one of its K-nearest

Figure 12: Average K-discrepancy score for K=1, 5, and 10, in the training data. The histogram shrinks quickly as K grows, which shows that, within a given cohort, for each hotel either the hotel itself or another similar hotel has a low discrepancy score.

neighbors has a low ranking discrepancy. Thus, for these hotels, we have sufficient data to determine how the demand for that hotel would change if the algorithm were to be scaled up. In Figure 12, we plot the average K-discrepancy score for the hotels in our data for K = 1, 5, and 10. The histograms in Figure 12 shrink quickly as K increases, which shows that for most hotels, there exist reasonably similar hotels that have low discrepancy.

In sum, we have a large number of low-discrepancy observations both at the: (1) within-hotel across-cohort level and (3) within-cohort but across-hotel level, to be able to estimate TATE reliably.

8 Counterfactual Analysis

To demonstrate the effectiveness of our method, we estimate the TATE of the Genesis algorithm based on Equation (11) and our empirical estimates of two models for clicks and impressions when the discrepancy vector ($\mathbf{d}^a_{jc_t}$) is set to zero. As discussed in §4.2, our counterfactual analysis relies on two assumptions. First, figures/ML/distr_query-eps-converted-to.pdf

Figure 13: Realized click-rate for the Genesis algorithm under the A/B test versus predicted TATE for Genesis, that is the click-rate when the Genesis algorithm is assigned to all users on the platform.

we assume that the total number of queries on any given day (Q_t) and the composition of queries (defined by the seasonality vector $\mathbf{g}_t$) generated at the platform do not change under different algorithms. This is reasonable since, in the short term, a user's decision to query and the destination of her query (e.g., Madrid) are unlikely to be influenced by the algorithm used by the platform (especially since users have no visibility into the algorithms they are assigned to). That said, note that the algorithm is allowed to affect the way users interact with the query results and influence their purchase decisions (e.g., what hotels are shown and whether they are clicked/purchased), i.e., all the users' actions following the query are allowed to be endogenous to the algorithm.

8.1 Counterfactual Estimates of TATE

We first present the observed and predicted TATE for the click-through rates for each day in the training and test data. To generate the predicted counterfactual TATEs, for each day t in the training and test data, we take the number of queries, and the hotel and seasonality features (Q_t, $\mathbf{x}_j$, $\mathbf{g}_t$) as given and simulate Clicks_t^a for 1000 draws at $\mathbf{d}_{jc_t}^a = \mathbf{0}$. For each draw, we then derive the click-through rate as the simulated click divided by Q_t and present the results from this exercise in Figure 13. See Appendix D for details of the exact counterfactual simulation procedure.

We first focus on the TATE predictions for the training data (i.e., cohorts 1–15) in Figure 13 and provide three main takeaways. First, our counterfactuals suggest that the expected performance of Genesis when we scale it up to 100% is systematically different from its observed performance throughout the training data (as depicted by the red triangles). This finding confirms the presence of interference bias and suggests that naively extrapolating based on the observed performance can be misleading. Second, we find that the counterfactual estimate of TATE remains pretty stable as we move from one cohort to another. This is in contrast to the observed performance, which varies a lot as we move across cohorts because the extent of interference bias varies in each cohort (due to shifts in the set of algorithms that are being tested and their proportions).

While we have theoretically established that our method recovers the true TATE in §4.3, it is useful to see some empirical evidence as well. Ideally, if we had access to a cohort where Genesis was fully deployed on the entire platform, that would serve as the ground truth against which we could compare our predicted counterfactual TATE estimates. Unfortunately, we do not observe any such cohort in our data. However, we can present the following evidence to corroborate the validity of the counterfactual TATE estimates within the scope of our data:

- **In-sample validation:** One interesting observation is that in cohort 8 (where 90% of the users were assigned to Genesis), we see that the predicted TATE by our model and the observed performance are very close. Thus, this specific cohort does not seem to suffer from significant interference bias. This corroborates the fact that as we scale up the portion of users assigned to a given algorithm, the observed performance of that algorithm approaches the true TATE. That said, it does not follow that cohorts with a higher fraction of users assigned to Genesis will necessarily have lower interference bias because the extent of interference bias also depends on the extent to which the other algorithms being tested are similar to Genesis. Overall, both the set of algorithms in a cohort as well their proportions can affect the discrepancy in rankings and the extent of interference.[22]
- **Out-of-sample validation:** Note that the last cohort (cohort 16) was not used for training. Also, going from cohort 15 to 16, the Genesis algorithm was scaled up from 60 to 80 percent, and its performance

[22] For example, suppose that we are interested in the performance of algorithm A and we have two cohorts, where we test A against another algorithm. Suppose cohort one consists of algorithms A and B in ratios 70–30, and cohort two consists of algorithm A and another algorithm C assigned in ratios 50–50. Further, suppose that algorithm B is very different from A while algorithm C is very similar to A. Even though the fraction of users assigned to A is lower in the second cohort, the extent of interference and bias in the observed performance would be lower than in the first cohort.

Cohort	Genesis share	Predicted Click-through Rate			Observed	Relative
		1^{st} Percentile	99^{th} Percentile	Mean	Click-through Rate	Bias
15	60%	0.460	0.462	0.461	0.531	15.18%
16	80%	0.493	0.495	0.494	0.475	-3.85%

Table 6: Predicted and observed click-through rates under the Genesis algorithm across cohort 15 and 16.

dropped discontinuously under the A/B test, see red triangles in Figure 13. Interestingly, the predicted TATE moves continuously across the two cohorts and is closer to the measurements made in cohort 16, where the algorithm was scaled up, see blue dots in Figure 13. In the next section, we investigate the performance of the algorithm across the last two cohorts to provide further evidence for the effectiveness of our algorithm in recovering TATE.

8.2 Comparison of Last Two Cohorts

We now simulate cohort-level distributions of TATE predictions and compare them to the observed TATE in Table 6 for cohorts 15 and 16. The details of simulation procedure used to produce the results shown in Table 6 are described in Appendix D. We find that the in-sample predictions of the true TATE in cohort 15 are lower than the observed CTR in this cohort (see the first row). The predicted true TATE for cohort 15 (0.461) is closer to the realized performance in cohort 16 (0.475) than the realized performance in cohort 15 (0.531). We now define Relative Bias as:

$$\text{Relative Bias} = \frac{\text{Predicted TATE - Observed TATE}}{\text{Observed TATE}}, \tag{14}$$

where the predicted and observed TATEs are defined in Equations (5) and (6), respectively. These two constructs denote our prediction of the true TATE and the realized performance of the algorithm.

We find that the relative bias in cohort 15 is equal to $\frac{0.531-0.461}{0.461} = 15.18\%$, while the relative bias in the last cohort is smaller at $\frac{0.475-0.494}{0.494} =$ -3.85%.[23] Notice that, while the observed performance suffers a discontinuous drop going from cohort 15 to 16, the predicted TATE is similar across the two cohorts and is closer to the realized performance in cohort 16. This is expected because the fraction of users allocated to Genesis is scaled up in the last cohort (to 80%), and as such, the extent of interference bias is likely to be small in this cohort. Overall, these findings are reassuring because our algorithm did not have access to the hold-out sample (cohort 16) but could still generate a reasonable estimate for TATE within this cohort.

8.3 Quantifying the Economic Impact of Interference Bias

So far, all our analysis focused on demand as measured by clicks, i.e., predicting the true TATE for clicks. In this section, we expand our analysis to include booking (i.e., revenues) to examine the economic impact of

[23] Note that we do not observe the ground truth, and the bias here is defined as the percentage difference between observed TATE under the A/B test, and the TATE predicted by our algorithm. As we would expect, the relative bias shrinks in cohort 16 relative to cohort 15 as the algorithm is scaled up.

Cohort	Genesis share	Predicted Booking Rate 1^{st} Percentile	 99^{th} Percentile	 Mean	Observed Booking Rate	Relative Bias
15	60%	0.0196	0.0201	0.0198	0.0257	29.8%
16	80%	0.0208	0.0212	0.0210	0.0228	8.57%

Table 7: Predicted and observed booking rates under the Genesis algorithm across cohort 15 and 16.

interference bias on firm's decisions. To that end, we build on Equation (11) to model bookings as follows:

$$\begin{aligned}\mathbb{E}[\text{Book}_t^a] = Q_t^a \cdot \sum_{j=1}^{J} \big[&\mathbb{P}[\text{Imp}|\text{Query}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t} = \mathbf{0}] \\ &\cdot \mathbb{P}[\text{Clicks}|\text{Imp}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t} = \mathbf{0}] \cdot \mathbb{P}[\text{Book}|\text{Click}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t} = \mathbf{0}]\big] .\end{aligned} \tag{15}$$

Recall that we already have the models for $\mathbb{P}[\text{Imp}|\text{Query}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t} = \mathbf{0}]$ and $\mathbb{P}[\text{Click}|\text{Imp}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t} = \mathbf{0}]$. So we only need to learn $\mathbb{P}[\text{Book}|\text{Click}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t} = \mathbf{0}]$. For this purpose, we build a third deep neural network model that predicts bookings conditional on clicks. The objective function, feature sets, and architecture for this model are similar to those used in the models for impressions and clicks. Then, using all the three models, we conduct counterfactual simulations and compare the realized bookings and predicted bookings for the last two cohorts. The results from this exercise are shown in Table 7. Please see Appendix E for details on the booking model and the simulation procedure.

The results in Table 7 are similar to those in Table 6, i.e., the bias is much smaller in the last cohort where a larger portion of users was assigned to Genesis. Particularly the relative bias in cohort 15 is equal to $\frac{0.0257-0.0198}{0.0198} = 29.8\%$ versus $\frac{0.0228-0.021}{0.021} = 8.57\%$ in the last cohort. Similar to the patterns documented before, the TATE changes much less than the realized outcome when moving from one cohort to another. Further, the in-sample fit in cohort 15 suggests that TATE must be smaller than the realized booking rate, consistent with the fact that the realized booking rate shrinks when moving from cohort 15 to 16.

In sum, these findings suggest that simply extrapolating from a naive A/B test that suffers from interference bias can lead the firm to misestimate revenues by around 30%. In the context we look at, the firm would overestimate revenues. In contrast, if the firm uses our procedure to correct for the bias, it can significantly reduce the extent of bias in revenue estimates. Overall, our findings emphasize the importance of recognizing and correcting for interference bias in platforms that use ranking algorithms to sell products.

9 Conclusion

Digital marketplaces and platforms routinely conduct a wide range of A/B tests to identify optimal marketing policies. A/B tests play a central role in helping firms determine the optimal ranking ordering of items on their websites. While firms have become increasingly sophisticated in implementing A/B tests that determine in real-time the impact of ranking algorithms on their clicks or sales, one issue associated with such tests is interference bias. Interference can stem from both direct and indirect sources, e.g., feedback loops built into algorithms, supply-side constraints, and the strategic response by the sellers of the items listed.

We formally show that A/B tests of ranking algorithms can produce biased estimates of an algorithm's performance since they fail to simulate the counterfactual market equilibrium realized when that algorithm is fully deployed. We present a framework to recover the true TATE of an algorithm based on market-equilibrium concepts. We theoretically show that, under some regularity conditions, we can obtain the true TATE of an algorithm by isolating and pooling observations across multiple cohorts of experiments. Further, we introduce the concept of rank discrepancy and explain how this metric can identify items close to the counterfactual market equilibrium under the algorithm of interest. Our solution concept has two key strengths. First, it is versatile since it is agnostic to the source of the interference bias. Second, it is also agnostic to the exact nature of the algorithms being tested. We do not need to know how any algorithm in the A/B test actually does the ranking; we just need to see the resulting ranking distributions. We apply our framework to data from a large travel website and empirically establish the presence of interference bias. Finally, we demonstrate how our theoretical solution concept can be translated to a real empirical setting and develop a scalable machine learning algorithm to recover the true TATE of one particular algorithm of interest for this website.

We now discuss the implications of our work for firms. Our analysis suggests that interference effects can significantly bias the TATE estimates of ranking algorithms, even in fully randomized A/B tests. In our empirical setting, a naive interpretation of A/B test results led the firm to overestimate the true TATE of the algorithm of interest (Genesis) by 15% (on clicks) and 30% (on bookings). Note that this is not only a "misestimation" issue but can also compromise the firm's decision-making and lead the firm to choose a sub-optimal ranking algorithm (which in turn would lead to lower clicks/revenues). For instance, as we saw in Figure 6b, the performance of all three algorithms (1, 2, and Genesis) are similar in cohort 15. However, the performance of Genesis drops in cohort 16 (where it is scaled-up) compared to algorithms 1 and 2. Based on the data from cohort 15, the firm may be indifferent between the three ranking algorithms and choose to stick with Genesis as their default. However, the data from cohort 16 suggests that such a decision could be sub-optimal since the Genesis algorithm is strictly dominated by algorithms 1 and 2 in cohort 16.[24] As such, we caution firms to be careful in interpreting results they obtain from A/B tests on optimal rankings. Instead, firms should carefully evaluate the extent to which their naive TATE estimates may be biased. Then they should use existing data from previous A/B tests to de-bias their TATE estimates of all the algorithms of interest and base their decisions on the de-biased estimates.

Finally, our approach for bias correction has the practical advantage of not requiring firms to employ complicated randomization strategies or alter their testing infrastructure. Instead, firms can apply existing experimental methods and use pre-existing experimental data. Our paper also provides some simple testing recommendations to firms – in the presence of interference problems, firms should run a lot of A/B tests and focus on increasing the variation in product rankings *across* tests even if no one A/B test is perfect, which in turn would allow them to use our approach for bias reduction. We expect our theoretical and empirical contributions to be of interest to both academics and practitioners and spur additional research in this area.

[24] Of course, the performance of algorithm 1 and 2 could also suffer from interference bias, and the results in cohort 16 do not guarantee that they will outperform Genesis when scaled up.

References

P. Bajari, B. Burdick, G. Imbens, I. Rosen, T. Richardson, and J. McQueen. Multiple randomization designs for interference, 2020. URL https://assets.amazon.science/c1/94/0d6431bf46f7978295d245dd6e06/double-randomized-online-experiments.pdf.

T. Blake and D. Coey. Why marketplace experimentation is harder than it seems: The role of test-control interference. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 567–582, 2014.

I. Chakraborty, M. Kim, and K. Sudhir. Attribute sentiment scoring with online text reviews: Accounting for language structure and missing attributes. 2021.

O. Chapelle, T. Joachims, F. Radlinski, and Y. Yue. Large-scale validation and analysis of interleaved search evaluation. *ACM Transactions on Information Systems (TOIS)*, 30(1):1–41, 2012.

Y. Chen and J. F. Canny. Recommending ephemeral items at web scale. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 1013–1022, 2011.

B. De los Santos and S. Koulayev. Optimizing click-through in online rankings with endogenous search refinement. *Marketing Science*, 36(4):542–564, 2017.

P. Dhillon and S. Aral. Modeling dynamic user interests: A neural matrix factorization approach. *arXiv preprint arXiv:2102.06602*, 2021.

D. Dzyabura and R. Peres. Visual elicitation of brand perception. *Available at SSRN 3496538*, 2019.

D. Eckles, B. Karrer, and J. Ugander. Design and analysis of experiments in networks: Reducing bias from interference. *Journal of Causal Inference*, 5(1), 2017.

K. Falk. *Practical recommender systems*. Simon and Schuster, 2019.

A. Fradkin. A simulation approach to designing digital matching platforms. *Boston University Questrom School of Business Research Paper Forthcoming*, 2019.

A. Ghose, P. G. Ipeirotis, and B. Li. Examining the impact of ranking on consumer behavior and search engine revenue. *Management Science*, 60(7):1632–1654, 2014.

C. A. Gomez-Uribe and N. Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):1–19, 2015.

V. Ha-Thuc, A. Dutta, R. Mao, M. Wood, and Y. Liu. A counterfactual framework for seller-side a/b testing on marketplaces. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2288–2296, 2020.

D. Holtz and S. Aral. Limiting bias from test-control interference in online marketplace experiments. *Available at SSRN 3583596*, 2020.

D. Holtz, R. Lobel, I. Liskovich, and S. Aral. Reducing interference bias in online marketplace pricing experiments. *arXiv preprint arXiv:2004.12489*, 2020.

G. W. Imbens and D. B. Rubin. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press, 2015.

K. Jamieson and L. Jain. A bandit approach to multiple testing with false discovery control. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 3664–3674, 2018.

D. Jannach and M. Jugovac. Measuring the business value of recommender systems. *ACM Transactions on Management Information Systems (TMIS)*, 10(4):1–23, 2019.

T. Joachims, A. Swaminathan, and T. Schnabel. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 781–789, 2017.

R. Johari, H. Li, I. Liskovich, and G. Weintraub. Experimental design in two-sided platforms: An analysis of bias. *arXiv preprint arXiv:2002.05670*, 2020a.

R. Johari, L. Pekelis, and D. Walsh. Always valid inference: Continuous monitoring of A/B tests. *Operations Research*, 2020b.

J. Katukuri, T. Könik, R. Mukherjee, and S. Kolay. Recommending similar items in large-scale online marketplaces. In *2014 IEEE International Conference on Big Data (Big Data)*, pages 868–876. IEEE, 2014.

R. Kohavi, D. Tang, and Y. Xu. *Trustworthy online controlled experiments: A practical guide to a/b testing*. Cambridge University Press, 2020.

J. Liu, O. Toubia, and S. Hill. Content-based model of web search behavior: An application to tv show search. *Management Science*, 2021.

C. F. Manski. Identification of treatment response with social interactions. *The Econometrics Journal*, 16(1): S1–S23, 2013.

P. Nandy, K. Basu, S. Chatterjee, and Y. Tu. A/b testing in dense large-scale networks: design and inference. *arXiv preprint arXiv:1901.10505*, 2019.

O. Netzer, R. Feldman, J. Goldenberg, and M. Fresko. Mine your own business: Market-structure surveillance through text mining. *Marketing Science*, 31(3):521–543, 2012.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

D. B. Rubin. Formal mode of statistical inference for causal effects. *Journal of statistical planning and inference*, 25(3):279–292, 1990.

M. Saveski, J. Pouget-Abadie, G. Saint-Jacques, W. Duan, S. Ghosh, Y. Xu, and E. M. Airoldi. Detecting network effects: Randomizing over randomized experiments. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1027–1035, 2017.

R. M. Ursu. The power of rankings: Quantifying the effect of rankings on online consumer search and purchase decisions. *Marketing Science*, 37(4):530–552, 2018.

H. Yoganarasimhan. Search personalization using machine learning. *Management Science*, 66(3):1045—-1070, 2020.

Web Appendix

A Appendix for §4.3

A.1 Proof for Proposition 1

We first start with a Lemma on the continuity of the demand function.

Lemma 1. *For each hotel with features* $\mathbf{x}$ *the overall demand function* $q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot))$ *is a continuous function of* $\mathbf{x}_j, P_j^{\mathbf{m}_c}(\cdot), P_j^a(\cdot)$

Proof. Note that:

$$q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) = \sum_{r=1}^{\infty} P_j^a(r)\gamma_r \alpha(\mathbf{x}_j, \mathcal{S}_j(\mathbf{x}_j, P_j^{\mathbf{m}_c})),$$

From Assumption 1 we know that $S_j(\mathbf{x}_j, P_j^{\mathbf{m}_c})$ is a unique and a continuous function of $\mathbf{x}_j$ and $P_j^{\mathbf{m}_c}$. Therefore, the composition of $\alpha(.,.)$ and $S(.)$, which is $\alpha(\mathbf{x}_j, \mathcal{S}_j(\mathbf{x}_j, P_j^{\mathbf{m}_c}))$ is also a continuous function of $\mathbf{x}_j$ and $P_j^{\mathbf{m}_c}$. Finally, the term $\sum_{r=1}^{\infty} P_j^a(r)\gamma_r$ is just a constant, which implies that $q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) = \sum_{r=1}^{\infty} P_j^a(r)\gamma_r \alpha(\mathbf{x}_j, \mathcal{S}_j(\mathbf{x}_j, P_j^{\mathbf{m}_c}))$ a continuous function of $\mathbf{x}_j$ and $P_j^{\mathbf{m}_c}$. □

Now, we present the proof of Proposition 1

Proof. We need to show that $\forall\ \epsilon > 0, \exists\ \delta > 0$, such that $\|P_j^a - P_j^{\mathbf{m}_c}\|_1 < \delta$ implies:

$$\left| q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) - q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot)) \right| < \epsilon.$$

From Lemma 1 we know that $q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot))$ is a continuous function of its arguments, therefore, $\exists\ \delta(\epsilon) > 0$ such that:

$$\|\{\mathbf{x}_j, P_j^a, P_j^{\mathbf{m}_c}\} - \{\mathbf{x}'_j, P'^a_j, P'^{\mathbf{m}_c}_j\}\|_1 < \delta(\epsilon)$$

ensures that:

$$\left| q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) - q_a^{(d)}(\mathbf{x}'_j; P'^a_j(\cdot), P'^{\mathbf{m}_c}_j(\cdot)) \right| < \epsilon,$$

setting $\delta < \delta(\epsilon)$:

$$\|P_j^a - P_j^{\mathbf{m}_c}\|_1 < \delta \Rightarrow \left\| \{\mathbf{x}_j, P_j^a, P_j^{\mathbf{m}_c}\} - \{\mathbf{x}_j, P_j^a, P_j^a\} \right\|_1 < \epsilon$$

which ensures that:

$$\left| q_a^{(d)}(\mathbf{x}_j; P_j^a, P_j^{\mathbf{m}_c}) - q_a^{(d)}(\mathbf{x}_j; P_j^a, P_j^a) \right| < \epsilon,$$

and completes the proofs. Note that in the above notation $P_j^{\mathbf{m}_c}$ and P_j^a are probability mass functions which can be regarded as infinite dimensional vectors in ℓ_1 as the integral of PMF functions is finite and is equal to one. □

A.2 Proof of Proposition 2

Proof. Part (1) follows directly from Proposition 1. Part (2) is also direct consequence of Lemma 1. Since $q_a^{(d)}(\cdot)$ is continuous, $\exists\ \delta(\epsilon) > 0$ such that:

$$\|\{\mathbf{x}_j, P_j^a, P_j^{\mathbf{m}_c}\} - \{\mathbf{x}'_j, P_j^{'a}, P_j^{'\mathbf{m}_c}\}\|_1 < \delta(\epsilon)$$

ensures that:

$$\left| q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) - q_a^{(d)}(\mathbf{x}'_j; P'^a_j(\cdot), P'^{\mathbf{m}_c}_j(\cdot)) \right| < \epsilon,$$

set $\delta = \frac{\delta(\epsilon)}{3}$ and let $\|P_{j'}^a - P_{j'}^{\mathbf{m}_c}\|_1 < \delta$, $\|P_{j'}^a - P_j^a\|_1 < \delta$, and $\|\mathbf{x}_j - \mathbf{x}_{j'}\|_1 < \delta$, then we have:

$$\|\{\mathbf{x}_{j'}, P_{j'}^a, P_{j'}^{\mathbf{m}_c}\} - \{\mathbf{x}_j, P_j^a, P_j^{\mathbf{m}_c}\}\|_1 < 3 \cdot \delta = \delta(\epsilon),$$

which implies that:

$$\left| q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) - q_a^{(d)}(\mathbf{x}_{j'}; P_{j'}^a(\cdot), P_{j'}^{\mathbf{m}_c}(\cdot)) \right| < \epsilon,$$

and completes the proof. □

A.3 Proof of Proposition 3

First, note that if $\|P_j^a - P_j^{\mathbf{m}_c}\|_1 \to 0$ then by Proposition 1 we have:

$$q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) \to q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot)).$$

Note that $\forall\ \epsilon > 0$

$$\exists c\ \in C \quad \text{such that} \quad \|P_j^a - P_j^{\mathbf{m}_c}\| < \epsilon,$$

which implies that for each hotel j there exists a sub-sequence of cohorts c_{j_n} such that $\|P_j^a - P_j^{\mathbf{m}_{c_{jn}}}\|_1 \to 0$ as $n \to \infty$. Then proposition 1 implies that:

$$q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_{c_{jn}}}(\cdot)) \to q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot)). \tag{A1}$$

Similarly, if $\forall \epsilon > 0$:

$$\exists\ j' \in J, c \in C \quad \text{such that} \quad \max\{\|\mathbf{x}_j - \mathbf{x}_{j'}\|, \|P_j^a - P_{j'}^a\|, \|P_{j'}^a - P_{j'}^{\mathbf{m}_c}\|\} < \epsilon,$$

then there exists a sub-sequence of cohorts c_{j_n} and a hotel j' such that

$$\max\{\|\mathbf{x}_j - \mathbf{x}_{j'}\|, \|P_j^a - P_{j'}^a\|, \|P_{j'}^a - P_{j'}^{\mathbf{m}_{c_{jn}}}\|\} \to 0 \quad \text{as} \quad n \to \infty,$$

and Proposition 2 implies that:

$$q_a^{(d)}(\mathbf{x}_{j'}; P_{j'}^a(\cdot), P_{j'}^{\mathbf{m}_{c_{jn}}}(\cdot)) \to q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot)). \tag{A2}$$

Therefore, for each hotel j there exists a sequence of estimates Y_{j_n} equal to the left hand side of either A1 or A2[25] which converges to the true demand for that hotel as $n \to \infty$, which implies that:

$$\sum_{j \in J} Y_{j_n} \to \sum_{j \in J} q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^a(\cdot)).$$

The latter implies that the true TATE can be constructed as the limit of $\sum_{j \in J} Y_{j_n} \to \sum_{j \in J}$ which completes the proof.

[25] Depending on the condition that holds for each hotel either a sequence of demand estimates for that same hotel or a sequence of estimates for a hotel similar j' to it converge to its true demands.

A.4 Solution and Proofs for General Case with both Direct and Indirect Interference

The solution in the main text relied on the assumption of no direct interference. We now extend it to the case where we allow for both direct and indirect interference.

Let $P_j^{a_c}(\cdot)$ denote the distribution of rankings for an item j under algorithm a in cohort c, such that $P_j^{a_c}(\cdot)$ is a function of the overall distribution of rankings for that item ($P_j^{\mathbf{m}_c}(\cdot)$), the equilibrium state for that item ($\mathcal{S}_j$ at $\mathbf{m}_c$), and its features ($\mathbf{x}_j$). Because the distribution of rankings under algorithm a can be influenced by other algorithms in cohort c, we now use the notation $P_j^{a_c}(\cdot)$ to denote this distribution (unlike $P_j^a(\cdot)$ in the main text).

As discussed in §4.1, the equilibrium state $\mathcal{S}_j$ for item j only depends on the overall distribution of rankings and its features $\mathbf{x}_j$. Therefore, without loss of generality, let us assume that for any item j with feature vector $\mathbf{x}_j$ there exists a deterministic function $f_{\mathbf{x}_j} : \ell^1 \to \ell^1$ that maps the overall distribution of rankings in the cohort to the ranking distribution returned for item j under algorithm such that: $P_j^{a_c}(\cdot) = f_{\mathbf{x}_j}(P_j^{\mathbf{m}_c}(\cdot))$. This implies that the ranking distribution of each item in algorithm a only depends on the item's own overall ranking and not on the outcomes of other items.

Let $\tilde{P}_j^a$ be the fixed point for $f_{\mathbf{x}_j}(.)$. Our goal is to estimate $q_a^{(d)}(\mathbf{x}_j; \tilde{P}_j^a(\cdot), \tilde{P}_j^a(\cdot))$, However, in the data we only observe $q_a^{(d)}(\mathbf{x}_j; P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot))$ for the cohorts where a was a participating algorithm. As before, the idea is to use observations from cohorts where $\|P_j^{\mathbf{m}_c}(\cdot) - P_j^{a_c}(\cdot)\|$ is sufficiently small to recover $q_a^{(d)}(\mathbf{x}_j; \tilde{P}_j^a, \tilde{P}_j^a)$. For this approach to work we need to ensure that: (a) $f_{\mathbf{x}_j}(.)$ has a unique fixed point , $\tilde{P}_j^a$, and (b) as $\|P_j^{\mathbf{m}_c}(\cdot) - P_j^{a_c}(\cdot)\| \to 0$, then $q_a^{(d)}(\mathbf{x}_j; P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot)) \to q_a^{(d)}(\mathbf{x}_j; , \tilde{P}_j^a(\cdot), \tilde{P}_j^a)(\cdot))$. As such, we need the following additional assumption.

Assumption 3. *For each item j with features $\mathbf{x}_j$ in a cohort c, there exists a function $f_{\mathbf{x}_j} : \ell^1 \to \ell^1$ such that $P_j^{a_c} = f_{\mathbf{x}_j}(P_j^{\mathbf{m}_c})$, $f_{\mathbf{x}_j}(.)$ is Lipschitz with Lipschitz constant $\lambda < 1$.*

This assumption has two implications. First, the rank distribution of an item j under algorithm a in cohort c, $P_j^{a_c}$, is assumed to only depend on j's own ranking in the cohort. That is, it is not allowed to be a function of other items' ranking in the cohort. Second, it ensures that $f_{\mathbf{x}_j}(.)$ is a contraction mapping. Since ℓ^1 with the absolute-value norm is complete, the Banach fixed point theorem implies that its fixed point is unique (Banach, 1922). The uniqueness of the fixed point is important because throughout the panel whenever the discrepancy of a focal algorithm ranking and the cohort ranking distributions for a given item is small then those observations are close to a unique fixed point of $f_{\mathbf{x}_j}(.)$ and we need not worry about data points with small discrepancy between $P_j^{a_c}$ and $P_j^{\mathbf{m}_c}$ being located in neighborhood of multiple fixed points.

Under Assumptions 2 and 3, we know that $\|P_j^{\mathbf{m}_c}(\cdot) - P_j^{a_c}(\cdot)\| \to 0$, we have $q_a^{(d)}(\mathbf{x}_j; P_j^a(\cdot), P_j^{\mathbf{m}_c}(\cdot)) \to q_a^{(d)}(\mathbf{x}_j; \tilde{P}_j^a(\cdot), \tilde{P}_j^a(\cdot))$. This is important because both $P_j^{a_c}(\cdot)$ and $P_j^{\mathbf{m}_c}(\cdot)$ are *equilibrium outcomes* and a priori we do not know how they would change as an algorithm is scaled up. Nevertheless, with these two assumptions if we ensure that $\|P_j^{\mathbf{m}_c}(\cdot) - P_j^{a_c}(\cdot)\| \to 0$ we know that we are approaching the *unique* equilibrium. Thus, as in the case of the simple solution, we can leverage the variation across cohorts and items to identify observations with low discrepancy, and infer the TATE from such observations.

Proposition 4. *Let J and C be the set of items and cohorts, respectively. Then, one can recover the TATE for all items $j \in J$, if the number of cohorts $|C| \to \infty$ and $\forall \epsilon > 0, \forall j \in J$ either*

$$\exists c \in C \quad \textit{such that} \quad \|P_j^{a_c} - P_j^{\boldsymbol{m}_c}\| < \epsilon,$$

$$\textit{or}, \ \exists \ j' \in J, c \in C \quad \text{such that} \quad \max\{\|\mathbf{x}_j - \mathbf{x}_{j'}\|, \|P_j^{a_c} - P_{j'}^{a_c}\|, \|P_{j'}^{a_c} - P_{j'}^{\mathbf{m}_c}\|\} < \epsilon.$$

Proof. See below. □

Proposition 4 is analogous to Proposition 3 in the main text. Essentially, it states that even in general case, with sufficient variation in ranking discrepancy between algorithm and cohort rankings for each hotel, one can recover the true TATE for each hotel as the number of cohorts grows.

We now present the proof of Proposition 4. To do so, we first start with two lemmas.

Lemma 2. *Let $f : \ell_1 \to \ell_1$ be a continuous function with Lipschitz constant $\lambda < 1$, then $\forall \epsilon > 0, \exists \delta > 0$ such that $\|\boldsymbol{x} - f(\boldsymbol{x})\|_1 < \delta$ ensures that $\|\boldsymbol{x} - \boldsymbol{x}^*\| < \epsilon$, where $\boldsymbol{x}^*$ is the unique fixed point $\boldsymbol{x}^* = f(\boldsymbol{x}^*)$.*

Proof. Using Banach (1922) we know that $f(.)$ has a unique fixed point $\mathbf{x}^*$. for a given $\mathbf{x}$ define $\mathbf{x}_0 = \mathbf{x}$, and $\mathbf{x}_n = f(\mathbf{x}_{n-1})$. The sequence $\mathbf{x}_n$ is Cauchy and converges to $\mathbf{x}^*$. Also we have:

$$\|\mathbf{x}_n - \mathbf{x}\|_1 = \left\|\sum_{i=0}^{n-1} \mathbf{x}_{i+1} - \mathbf{x}_i\right\|_1 \overset{(1)}{\leq} \sum_{i=0}^{n-1} \|\mathbf{x}_{i+1} - \mathbf{x}_i\| \overset{(2)}{\leq} \sum_{i=0}^{n-1} \lambda^i \|f(\mathbf{x}) - \mathbf{x}\| = \frac{1-\lambda^n}{1-\lambda}\|f(\mathbf{x}) - \mathbf{x}\| < \frac{1}{1-\lambda}\|f(\mathbf{x}) - \mathbf{x}\|,$$

where (1) follows from the triangle inequality, and (2) is a direct result of $f(.)$ being Lipschitz with Lipschitz constant $\lambda < 1$.

Since $\mathbf{x}_n$ converges to $\mathbf{x}^*$, $\forall \epsilon > 0, \exists N_\epsilon > 0$, such that $n > N_\epsilon$:

$$\|\mathbf{x}_n - \mathbf{x}^*\| < \frac{\epsilon}{2},$$

set $\delta < \frac{(1-\lambda)\epsilon}{2}$, and $N > N_\epsilon$ then we have:

$$\|\mathbf{x}_n - \mathbf{x}\|_1 < \frac{1}{1-\lambda}\|f(\mathbf{x}) - \mathbf{x}\| < \frac{1}{1-\lambda} \cdot \frac{(1-\lambda)\epsilon}{2} < \frac{\epsilon}{2},$$

and

$$\|\mathbf{x}_n - \mathbf{x}^*\| < \frac{\epsilon}{2}$$

combining the latter two with triangle inequality yields:

$$\|\mathbf{x} - \mathbf{x}^*\| \leq \|\mathbf{x} - \mathbf{x}_n\| + \|\mathbf{x}_n - \mathbf{x}^*\| = \epsilon.$$

Therefore, setting $\delta = \frac{(1-\lambda)\epsilon}{2}$ ensures that $\|\mathbf{x} - f(\mathbf{x})\| < \delta \Rightarrow \|\mathbf{x} - \mathbf{x}^*\| < \epsilon$. □

Lemma 3. *Under Assumptions 1-3, if $\|P_j^{\mathbf{m}_c}(\cdot) - P_j^{a_c}(\cdot)\| \to 0$, then*

$$q_a^{(d)}(\mathbf{x}_j; P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot)) \to q_a^{(d)}(\mathbf{x}_j; \tilde{P}_j^{a}(\cdot), \tilde{P}_j^{a}(\cdot)).$$

Proof. We need to show that $\forall \epsilon > 0, \exists \delta > 0$ such that $\|P_j^{\mathbf{m}_c}(\cdot) - P_j^{a_c}(\cdot)\|_1 < \delta$ ensures that:

$$\left\|\{\mathbf{x}_j, P_j^{a_c}, P_j^{\mathbf{m}_c}\} - \{\mathbf{x}_j, \tilde{P}_j^{a}, \tilde{P}_j^{a}\}\right\| < \epsilon$$

which combined with continuity of $q_a^{(d)}$ completes the proof.

Under assumption 3, we have $P_j^{a_c}(\cdot) = f_{\mathbf{x}_j}(P_j^{\mathbf{m}_c}(\cdot))$. Using 2 we know that $\|P_j^{a_c}(\cdot) - P_j^{\mathbf{m}_c}(\cdot)\|_1 = \|P_j^{a_c} - f_{\mathbf{x}_j}(P_j^{a_c})\|_1 < \frac{(1-\lambda)\epsilon}{8}$ ensures that $\|P_j^{a_c} - \tilde{P}_j^a\| < \frac{\epsilon}{4}$, where λ is the Lipschitz constant of $f_{\mathbf{x}_j}$. Then we have:

$$\Big\|\{\mathbf{x}_j, P_j^{a_c}, P_j^{\mathbf{m}_c}\} - \{\mathbf{x}_j, \tilde{P}_j^a, \tilde{P}_j^a\}\Big\| = \|P_j^{a_c}-\tilde{P}_j^a\|+\|P_j^{\mathbf{m}_c}-\tilde{P}_j^a\| \leq \|P_j^{a_c}-\tilde{P}_j^a\|+\|P_j^{\mathbf{m}_c}-P_j^{a_c}\|+\|P_j^{a_c}-\tilde{P}_j^a\|$$

$$\leq \frac{\epsilon}{4} + \frac{(1-\lambda)\epsilon}{8} + \frac{\epsilon}{4} < \epsilon$$

which completes the proof. □

Now we present the proof of Proposition 4.

If for $\forall\ j \ \in J$ and $\epsilon > 0$:

$$\exists c \in C \quad \text{such that} \quad \|P_j^{a_c} - P_j^{\mathbf{m}_c}\| < \epsilon,$$

then, for each hotel j there exists a sub-sequence of cohorts c_{j_n} such that:

$$\|P_j^{a_{c_{j_n}}} - P_j^{\mathbf{m}_{c_{j_n}}}\| \to 0 \quad \text{as} \quad n \to \infty,$$

then Lemma 3 implies that:

$$q_a^{(d)}(\mathbf{x}_j; P_j^{a_{c_{j_n}}}(\cdot), P_j^{\mathbf{m}_{c_{j_n}}}(\cdot)) {\rightarrow} q_a^{(d)}(\mathbf{x}_j; \tilde{P}_j^a(\cdot), \tilde{P}_j^a(\cdot)).$$

On the other hand, if for each j there exists a hotel j':

$$\exists\ j' \in J, c \in C \quad \text{such that} \quad \max\{\|\mathbf{x}_j - \mathbf{x}_{j'}\|, \|P_j^{a_c} - P_{j'}^{a_c}\|, \|P_{j'}^{a_c} - P_{j'}^{\mathbf{m}_c}\|\} < \epsilon, \tag{A3}$$

then, for each hotel j there exists a hotel j', and a sub-sequence of cohorts c_{j_n} such that:

$$\max\{\|\mathbf{x}_j - \mathbf{x}_{j'}\|, \|P_j^{a_c} - P_{j'}^{a_c}\|, \|P_{j'}^{a_c} - P_{j'}^{\mathbf{m}_c}\|\} \to 0$$

then an argument similar to the proof of Lemma 3 implies that:

$$q_a^{(d)}(\mathbf{x}_{j'}; P_{j'}^{a_{c_{j_n}}}(\cdot), P_{j'}^{\mathbf{m}_{c_{j_n}}}(\cdot)) {\rightarrow} q_a^{(d)}(\mathbf{x}_j; \tilde{P}_j^a(\cdot), \tilde{P}_j^a(\cdot)). \tag{A4}$$

Therefore, for each hotel j there exists a sequence of estimates Y_{j_n} equal to the left hand side of either A3 or A4[26], which converges to the true demand for that hotel as $n \to \infty$. This, in turn implies that:

$$\sum_{j \in J} Y_{j_n} \to \sum_{j \in J} q_a^{(d)}(\mathbf{x}_j; \tilde{P}_j^a(\cdot), \tilde{P}_j^a(\cdot)).$$

B Booking Regressions

In § 6.2 we documented that the change in click-through rate across cohorts 15 and 16 is more pronounced for hotels for which the overall discrepancy is high. In this section, we document similar patterns for booking rates.

[26] Depending on the condition that holds for each item, either a sequence of demand estimates for that same item or a sequence of estimates for an item similar j' to it converge to its true demand.

Instead of using the click-through rate as the dependent variable in Equation (10), we use the booking rate $\mathbf{Y}_{jc} = \frac{\text{1 + bookings for hotel j in cohort c}}{\text{1 + total impressions of hotel j in cohort c}}$. We report the results in Table A1. Column (1) in Table A1 suggests that hotel booking rate drops on average by $\exp(\text{-}0.604) - 1 = \text{-}45.34\%$ as we move from cohort 15 to cohort 16, an average that is calculated across all hotels. Column (2) shows that this drop in booking rate is mainly explained by the change in booking rates of hotels that had high overall rank discrepancy ($\exp(\text{-}0.949) - 1 = \text{-}61.28\%$) compared to a much lower common trend ($\exp(\text{-}0.192) - 1 = \text{-}17.46\%$). These patterns are consistent with those documented in Table 5.

	Dependent variable:		
	Log(Hotel booking rate)		
	(1)	(2)	(3)
Last cohort x High overall rank discrepancy		-0.949^{***}	-0.949^{***}
		(0.014)	(0.014)
Last cohort	-0.604^{***}	-0.192^{***}	-0.192^{***}
	(0.007)	(0.007)	(0.007)
High overall rank discrepancy		1.588^{***}	
		(0.014)	
Constant	-5.640^{***}	-6.321^{***}	
	(0.008)	(0.009)	
Hotel FE			X
Observations	85,292	85,292	85,292
R^2	0.037	0.179	0.801
Adjusted R^2	0.037	0.179	0.601
Residual Std. Error	1.540 (df = 85290)	1.422 (df = 85288)	0.991 (df = 42644)
Note:			*p<0.1; **p<0.05; ***p<0.01

All standard errors are clustered at hotel level.

Table A1: Change in booking rate of hotels with low/high overall ranking discrepancy across the last two cohorts.

C Appendix for Feature Sets

C.1 Hotel Features

As discussed in the main text, to predict the TATE of an algorithm, we can use variation in ranking discrepancy for similar hotels in the same cohort. For instance, if the ranking of a hotel A within a given algorithm is largely consistent with its cohort ranking (low discrepancy) then its realized demand in that cohort could be useful in determining the true demand for hotels similar to A which ended up having a higher ranking discrepancy between their algorithm and cohort rankings. Features such as latitude, longitude, included amenities, or user ratings are available for only a subset of hotels in our data and therefore cannot be used to reliably create hotel features for the entire set of hotels. Hence, to capture similarities across the hotels in our data we rely on the co-occurrence of hotels in search results and users' clicking behavior. Recall that our goal is to capture whether hotels are perceived to be similar for the eye of a given algorithm (Genesis in this case). The idea is that if two hotels are ranked similarly by Genesis, and one has low ranking discrepancy with the overall cohort and another one has high ranking discrepancy, then the performance of the hotel with low ranking discrepancy could inform us about how the performance of the hotel with high ranking discrepancy would change when the algorithm is scaled up. To capture this idea, we create a co-occurrence matrix for hotels $\mathbf{Z}$ with elements $z_{jj'}$ reporting the number of times that hotels j and j' appeared in the same result page together. We also consider the number of times where hotels j and j' were clicked on by the same user and create another matrix $\overline{\mathbf{Z}}$ with elements $\bar{z}_{jj'}$ that reflect the number of times hotels j and j' were clicked on by the same user in a given session. Intuitively, $\mathbf{Z}$ and $\overline{\mathbf{Z}}$ could reflect the similarity of two hotels.

Global Vectors (GloVe) is a widely used method for learning vector representations using word co-occurrences in the context of natural language processing (Pennington et al., 2014). We use GloVe to learn vector representations for the hotels in our data. The idea is to solve the following optimization problem:

$$\underset{\mathbf{W},\mathbf{b}}{\text{minimize}} \quad \sum_{j,j'=1}^{\mathcal{J}} f(z_{jj'}) \left(\mathbf{W}_j^T \mathbf{W}'_j + b_j + b'_j - \log z_{jj'}\right)^2, \tag{A5}$$

where $z_{jj'}$ is the number of times hotel j and j' appeared in the same result page. $\mathbf{W}$ is the representation matrix where its j^{th} row $\mathbf{W}_j$ is the representation vector for hotel j. The terms b_j and b'_j are added to absorb the baseline differences in frequency of hotel occurrences. Finally, $f(z_{jj'})$ is a weighting function that prevents the algorithm from focusing only on very common hotels and is defined as:

$$f(z_{jj'}) = \begin{cases} (\frac{z_{jj'}}{z_{max}})^{\psi} & \text{if } z_{jj'} < z_{max}, \\ 1 & \text{otherwise,} \end{cases} \tag{A6}$$

Where ψ is a parameter that determines the shape of the weighting function, we set $\psi = \frac{3}{4}$ similar to Pennington et al. (2014) and Mikolov et al. (2013). We set z_{max} to 10, and solve the optimization problem above once using the occurrence matrix $\mathbf{Z}$, learning a 25 dimensional representation vector for each hotel. We repeat this process for $\overline{\mathbf{Z}}$ as well to learn another set of 25 dimensional vector for hotels. Overall, this process yields a representation vector of length 50 for each hotel in our data.

To verify that the representation vectors we learnt do reflect similarities across hotels, we compare them for hotels located in six major destinations in our data, namely, Berlin, Paris, Madrid, London, Dubai, and Rome. Since comparing 50 dimensional vector representations is not straightforward, we use t-SNE (Van der Maaten and Hinton, 2008), which is a visualization technique that projects high dimensional vectors into a 2-D plane. The t-SNE projection of the 50 dimensional representation vectors for the hotels located across the six major cities in our data are plotted in Figure A1. As expected, hotels that are located in the same city are clustered next to each other in Figure A1, which reassures us that the representation method is functioning as expected.

Intuitively, the matrix $\mathbf{Z}$ is capturing how a given algorithm perceives different hotels, while $\overline{\mathbf{Z}}$ includes information about user preferences for different hotels. Note that unobservables such as commissions may be factored into ranking algorithms and affect a hotels ranking, but those features may be unrelated to user demand. Therefore, if the GloVe features extracted from $\mathbf{Z}$ and $\overline{\mathbf{Z}}$ are similar for two hotels j and j', it indicates that they are perceived as being similar both by Genesis and users on the platform. This means that the observed demand of j' in cohorts where the algorithm and cohort ranking discrepancy of j' is low are useful not only for detecting the true demand for j' under Genesis, but is also informative of the true demand of item j under Genesis.

C.2 Seasonality Features

In principle, is possible to use day or week dummies to capture seasonal trends.. However, that would mean that we can predict the TATE only in time periods that appear in the training data. Since our goal is to test figures/illustrative/hotel_representation_tsne-eps-converted-to.pdf

Figure A1: 2D t-SNE projection of hotels across six major destinations in data.

whether the model has the ability to predict the change in the click-rate in the last cohort (test data), we instead use a vector representation for each day that is calculated using the composition of queries submitted in that day. The main assumption is that the composition of queries are exogenous and do not depend on the search algorithm, i.e., the search algorithm does not change the likelihood of submitting a query for a give destination. The composition of queries contain information about the type of users that arrive at the platform based on the type of destinations, e.g., business or leisure, and could be helpful in detecting seasonal patterns that generalized beyond the period used for training. To create a representation vector for each day in the data, We follow this procedure: (i) we determine the top 5000 destinations searched in our data[27], (i) for each destination, we create a vector by counting the number of times it was searched during each day in our panel, (iii) we concatenate these vectors to create a matrix with the number of days as its rows and 5000 thousand columns that correspond to the destination, (iv) we calculate the principle components of this matrix, and (v) the top 25 principle components yield a 25 dimensional representation vector for each day in our data which is used as an input to our machine learning algorithm.

[27] The top 5000 destinations are responsible for 90% of queries submitted on the platform.

C.3 Ranking Discrepancy Features

In the preliminary analysis in §6.2, we used the log ratio of expected rankings to capture the discrepancy between the ranking distribution of a focal hotel under a given algorithm a and under the entire cohort c. This measure is effectively a simple distance measure between two distributions. However, it only focuses on the means of two ranking distributions. However, in practice, two distributions with the same mean can diverge significantly in reality. Therefore, we now augment our discrepancy metric by adding: (1) the log ratio of deciles across these two distributions, and (2) the log ratio of share of impressions between Genesis and the cohort. We discuss both these discrepancy metrics below.

First, the log ratio of the ranking at decile δ is defined as follows:

$$\xi_\delta(P_j^{a_c}(\cdot), P_j^{\mathbf{m}_c}(\cdot)) = \log\left(\frac{d_\delta(P_j^{a_c})}{d_\delta(P_j^{\mathbf{m}_c})}\right), \tag{A7}$$

where $d_\delta(.)$ denotes the δ^{th} decile of the distribution. This gives us ten log-ratios, and together, they can capture the discrepancy across the two distributions of rankings in a more complete fashopn. To illustrate the intuition behind this metric, we present Figure A2, which compares the deciles across two hypothetical empirical CDFs (which have the same mean). Notice that there is variation in the distance between corresponding deciles and our vector of discrepancy metrics should capture this variation.

Next, for each hotel we calculate the share of queries that the hotel appears in and add the log ratio of shares between the Genesis and the cohort as a discrepancy measure as well. Together, this gives us 12 discrepancy measures for each hotel-cohort combination, and they form our discrepancy vector $\mathbf{d}_{jc_t}^a$.

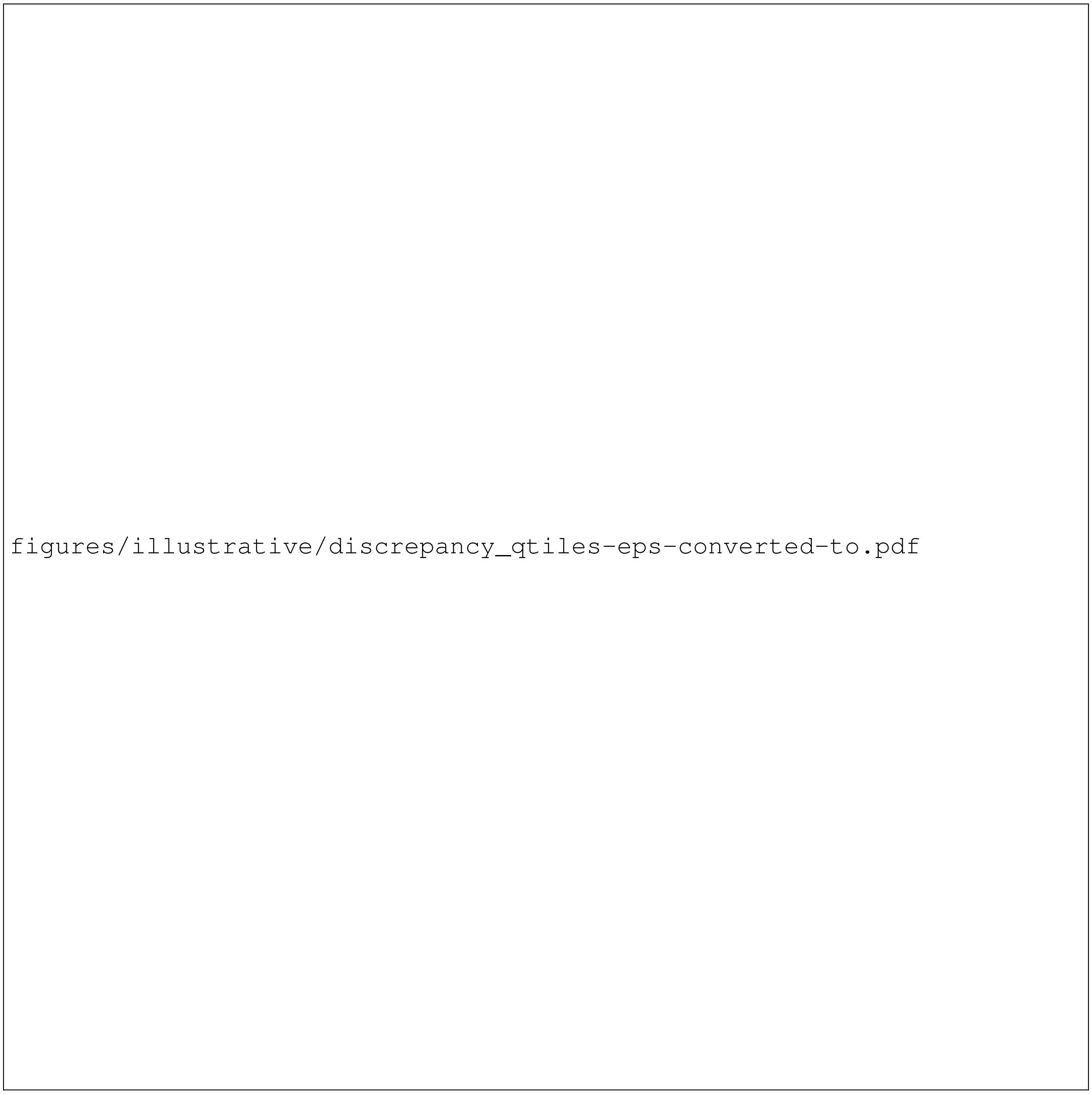

figures/illustrative/discrepancy_qtiles-eps-converted-to.pdf

Figure A2: The augmented discrepancy metric includes the log ratio of deciles across the distribution of Genesis and cohort rankings for each hotel as illustrated above.

D Procedure for Counterfactual Simulation

Our goal is to predict the click-through rate when algorithm a is scaled up to 100%, i.e., when the discrepancy between algorithm and cohort rankings is zero for all the items (i.e., $\forall\ \jmath \in J$). Recall that we trained two neural network models, one that models the number of impressions as a binomial random variable, and another one that models the number of clicks conditional on impressions, also as a binomial random variable. The outputs of the two neural networks are $p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$ and $p_1(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$, respectively, which denote the probability of click conditional on impression and probability of impression conditional on a query for a hotel with features $\mathbf{x}_j$ during a day with seasonality features $\mathbf{g}_t$ and discrepancy vector $\mathbf{d}^a_{jc_t}$), respectively.

We now provide a description of the counterfactual simulation. Algorithm 1 outlines the counterfactual simulation process. To predict the performance of algorithm a when discrepancy is zero, for each hotel-day observation we evaluate the output of the neural networks, that is p_0 and p_1, when $\mathbf{d}^a_{jc_t} = \mathbf{0}$.

The algorithm for the counterfactual simulation, with 1000 draws is given below:

Algorithm 1 Counterfactual sampling algorithm for click-through rate

```
for $t = 1$ to $T$ do                                                    ▷ t indexes days
    for $s = 1$ to 1000 do                                               ▷ s indexes samples
        $TotalImpressions_t^{(s)} \leftarrow 0$
        $TotalClicks_t^{(s)} \leftarrow 0$
        for $j = 1$ to $J$ do                                            ▷ j indexes hotels
            $P_0 \leftarrow p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{0})$
            $P_1 \leftarrow p_1(\mathbf{x}_j, \mathbf{g}_t, \mathbf{0})$
            Draw $Impressions \sim$ Binomial($Q_t$, $P_0$)               ▷ $Q_t$ is the total # of queries in day t
            Draw $Clicks \sim$ Binomial(Impressions, $P_1$)
            $TotalClicks_t^{(s)} \leftarrow TotalClicks_t^{(s)} + Clicks$
        end for
        $CTR_t^{(s)} \leftarrow \frac{TotalClicks_t^{(s)}}{Q_t}$
    end for
end for
```

E Booking Model and Counterfactuals

We now discuss the neural network model for calculating the booking rate. At a high-level this model is similar to the models that we built for impressions and clicks. The details are discussed below.

We start with the objective function for estimating the number of bookings for each hotel. Let $p_2(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$ be the probability that a hotel j with features $\mathbf{x}_j$ and algorithm-cohort discrepancy features $\mathbf{d}^a_{jc_t}$ is booked conditional on being clicked on a day characterized by the seasonality vector $\mathbf{g}_t$. We parameterize $p_2(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$ as a neural network similar to those described in §7. Further, let b^a_{jt} denote the number of bookings generated through the Genesis algorithm for a hotel j on a given day where $\mathcal{V}^a_{jt}$ denotes the number of clicks users made on the impressions to view the hotel in our data under the the Genesis algorithm during day t. Note that we keep the algorithm index a to remain consistent with notation developed in Section 4.1. Modeling this process as a binomial random variable, the log-likelihood of observing b^a_{jt} bookings for a hotel j, out of the $\mathcal{V}^a_{jt}$ clicks is given by:

$$\mathcal{L}(\mathcal{V}^a_{jt}, b^a_{jt}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t}; p_2) = \log\left[\binom{\mathcal{V}^a_{jt}}{b^a_{jt}} p_2(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})^{b^a_{jt}} \left(1 - p_2(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})\right)^{\mathcal{V}^a_{jt} - b^a_{jt}}\right]. \tag{A8}$$

Aggregating the above over all hotels and days in the training data, we have:

$$\mathcal{L} = \sum_{t=1}^{T} \sum_{j=1}^{J} \mathcal{L}(\mathcal{V}^a_{jt}, b^a_{jt}, \mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t}; p_2). \tag{A9}$$

We use a deep neural network to maximize the log-likelihood shown in Equation (A9) on our training data. The neural network architecture is similar to the ones used for learning clicks and impressions and schematic view is presented in Figure 10. For each hotel-day observation under the Genesis algorithm, the neural network outputs the probability of booking conditional on being clicked on, $p_2(\mathbf{x}_j, \mathbf{g}_t, \mathbf{d}^a_{jc_t})$, and tunes it to maximize the log-likelihood for the training data set.

Finally, to perform the booking counterfactuals, we follow the same steps described in Appendix D and append our booking model to the process. Algorithm 2 lays out the steps for sampling from the bookings. Once we sample bookings for each hotel under the counterfactual where discrepancy $\mathbf{d}^a_{jc_t} = \mathbf{0}$, we can aggregate to calculate the total bookings under an algorithm and then divide it by the total number of queries to get the bookings under the counterfactual where the algorithm is fully scaled.

Algorithm 2 Counterfactual sampling algorithm for booking rate

```
for $t = 1$ to $T$ do                                            ▷ t indexes days
    for $s = 1$ to 1000 do                                       ▷ s indexes samples
        $TotalImpressions_t^{(s)} \leftarrow 0$
        $TotalClicks_t^{(s)} \leftarrow 0$
        for $j = 1$ to $J$ do                                    ▷ j indexes hotels
            $P_0 \leftarrow p_0(\mathbf{x}_j, \mathbf{g}_t, \mathbf{0})$
            $P_1 \leftarrow p_1(\mathbf{x}_j, \mathbf{g}_t, \mathbf{0})$
            $P_2 \leftarrow p_2(\mathbf{x}_j, \mathbf{g}_t, \mathbf{0})$
            Draw $Impressions \sim$ Binomial($Q_t$, $P_0$)       ▷ $Q_t$ is the total # of queries in day t
            Draw $Clicks \sim$ Binomial(Impressions, $P_1$)
            Draw $Bookings \sim$ Binomial(Clicks, $P_2$)
            $TotalBookings_t^{(s)} \leftarrow TotalBookings_t^{(s)} + Bookings$
        end for
        $CTR_t^{(s)} \leftarrow \frac{TotalBookings_t^{(s)}}{Q_t}$
    end for
end for
```

References

S. Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fund. math*, 3(1):133–181, 1922.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.