A data-driven method for reconstructing a distribution from a truncated sample with an application to inferring car-sharing demand

Evan Fields$^1$, Carolina Osorio$^{1,2}$, and Tianli Zhou$^2$

$^1$Massachusetts Institute of Technology Operations Research Center
$^2$Massachusetts Institute of Technology Civil and Environmental Engineering

August 8, 2018

Abstract

This paper proposes a method to recover an unknown probability distribution given a censored or truncated sample from that distribution. The proposed method is a novel and conceptually simple detruncation method based on sampling the observed data according to weights learned by solving a simulation-based optimization problem; this method is especially appropriate in cases where little analytic information is available but the truncation process can be simulated. The proposed method is compared to the ubiquitous maximum likelihood (MLE) method in a variety of synthetic validation experiments where it is found that the proposed method performs slightly worse than perfectly specified MLE and competitively with slight misspecified MLE. The practical application of this method is then demonstrated via a pair of case studies in which the proposed detruncation method is used alongside a car-sharing service simulator to estimate demand for round trip car-sharing services in the Boston and New York metropolitan areas.

1 Introduction

Car-sharing is a transportation service in which travelers use cars that they do not own; the vehicles are thus shared between users of the car-sharing service. In this work, we focus on round-trip station-based car-sharing, which is one of the most common types of car-sharing worldwide (Barth and Shaheen 2002). In round-trip car-sharing, users must return the vehicle they use to the location where they picked up the vehicle; the alternatives are known as one-way service and free-floating service. ‘Station-based’ means vehicles are available for pickup and dropoff only at predesignated stations, unlike free-floating car-sharing where vehicles can be found or returned anywhere throughout a defined service region. In this paper, we use the term ‘location’ to refer to any place in a city, regardless of whether that place has any car-sharing vehicles, and the term ‘station’ refers to a location where round trip car-sharing vehicles can be picked up. Though car-sharing originated in Europe in the mid-20th century, it has only become widely popular in the last decades, especially in the United States (Jorge and Correia 2013, Bert et al. 2016).
As the popularity of car-sharing has grown, planning and operational concerns have led to interest in modeling the demand for car-sharing services. Users of car-sharing services frequently encounter capacity constraints: a potential user may not find an available reservation exactly matching the desired time, location, duration, and vehicle characteristics. In this case, a user can either accept a non-first-choice reservation or decide not to use the car-sharing service for the trip. In either case, the reservation that actually occurred—or lack thereof—does not accurately represent demand. To use common statistical language, the historical reservation data is truncated when a user abandons the system after failing to find an acceptable reservation and censored when a user accepts a sub-optimal reservation (Klein and Moeschberger 2005, Ch. 3). In this paper, for brevity we slightly abuse nomenclature and refer to any combination of censoring and truncation as ‘truncation’. Therefore, a car-sharing operator’s (CSO’s) historical reservation data represents a truncated sample of its users’ travel needs and preferences rather than the distribution of demand for car-sharing.

Though capacity constraints are ubiquitous in transportation and beyond, there are several reasons why the aforementioned truncation mechanisms are particularly relevant and difficult in round trip car-sharing: (i) at a given station and time, there are typically very few (often zero or one) vehicles available, so truncation is common; (ii) because all users of a car-sharing network share a common set of cars, each user’s reservations affect the vehicles available to the other users, especially because capacity is low, and therefore one incident of truncation can have effects that ripple outwards; (iii) because car-sharing trips are often discretionary trips in urban areas where other modal alternatives are available, potential users often can choose to make a trip without using the car-sharing network.

In many cases, a CSO would like to know the distribution of demand for car-sharing services. Such information is of interest, for example, for tactical decisions, such as how to spatially allocate the vehicles across the network of stations. The CSO may also be interested in inferring demand when planning network expansion; the CSO could identify characteristics common to stations with high demand, and then plan new stations which also share these characteristics. Estimating demand is also useful for a CSO making pricing decisions. For example, a CSO may have a target for what fraction of demand it should serve, and in conjunction with a pricing model, estimates of demand could be used to ensure that not too many potential users are priced out of the system. In all these cases, the data available to the CSO such as station-wise usage data may not accurately represent the demand, which is not directly observed. Researchers in many fields have addressed the problem of recovering a distribution given a censored or truncated sample from that distribution. Next, we highlight several popular approaches and describe both their advantages and their challenges in the car-sharing setting.

Maximum likelihood estimation (MLE) is the most common approach for recovering a latent (i.e. unobserved, but giving rise to observed data) distribution subject to known truncation (Talluri and Van Ryzin 2005, Ch. 9). A likelihood function for observed data is written by assuming a parametric form for the latent distribution and the truncation process, and then the maximum likelihood parameters are inferred. This approach is taken for example in the well-known vending machine study (Anupindi et al. 1998) where the unobserved arrival rate of customers is estimated by assuming a simple customer choice model along with a Poisson process for the customer arrival process. The popularity of maximum likelihood approaches is well deserved. MLE techniques are usually computationally efficient whether implemented by directly maximizing the log-likelihood or by
using the expectation-maximization algorithm (e.g. Bishop 2006, Ch. 9). In some cases, no maximization procedure is required because the maximum likelihood parameters can be computed in closed form. Further, the maximum likelihood estimator is known to have asymptotically optimal convergence properties (under mild regularity conditions, see e.g. Keener (2010, Ch. 9, 16)). Additionally, maximum likelihood estimation often performs well in practice. For a recent comparison of MLE to other inference methods, see Fridman and Lapina (2016).

However, MLE is not without its challenges. For example, the parameters maximizing the likelihood can correspond to a pathological or physically meaningless solution. Notably, this is a danger when fitting Gaussian mixture models: infinite likelihood can be attained by having one component of the mixture “collapse” towards a zero-variance singularity at a single data point (Bishop 2006, p434). In addition, this approach is ill-suited to the car-sharing case because the intricate interdependence of customers’ decisions makes writing an analytic likelihood impractical. To illustrate one facet of this interdependence, note that in a car-sharing context, it is common for multiple distinct reservations to be mutually exclusive. For example, the reservations “vehicle i from 2pm-4pm” and “vehicle i from 3pm-5pm” are distinct products for which there is likely separate demand, but if a customer takes one of these reservations, no other customer can choose the other. This introduces a complicated time and space dependency between reservations: the result of a user searching for a given reservation depends not just on whether that particular reservation is available, but also on the availability of all other reservations the user might accept. As a consequence, the order in which desired reservations arrive matters and thus reservations are not exchangeable; this significantly complicates writing a likelihood function which captures the reservation dynamics.

Maximum likelihood techniques typically require practitioners to incorporate specialized domain knowledge when assuming the parametric form of the latent distribution and truncation procedure. In the case of car-sharing, such knowledge is often not available. In our case study, for example, the a plausible parametric form of the distribution of car-sharing demand is not known a priori, and we are unable to make plausible distributional assumptions.

There has been some work on using nonparametric methods together with maximum likelihood; for example, Schafer (2007) infers a truncated bivariate density by assuming the only interaction effect between the variables is the simple product of those variables, though he also suggests using a physically motivated interaction term. This technique would be difficult to apply to the car-sharing demand distribution because demand depends on many factors including day of week, time of day, location, duration, and so forth. Therefore the number of potential interaction terms is large, and we don’t a priori know plausible forms for these interaction terms.

The problem of recovering a truncated distribution also occurs frequently in revenue management where it is often referred to as ‘unconstraining’: the number of sales of a product is constrained to be less than or equal to the stock of that product, and a retailer would like to know the unconstrained demand. See Talluri and Van Ryzin (2005, Ch. 9) for an overview of common unconstraining methods in revenue management, including an outline of how the expectation-maximization algorithm is commonly applied to maximum likelihood techniques for unconstraining. For a more recent high-level overview of unconstraining in revenue management, see Weatherford (2016). Occasionally, nonparametric unconstraining methods are used in revenue management. For example, the classic
Kaplan-Meier estimator (Kaplan and Meier 1958) is a nonparametric estimate of survival probabilities given censored survival data. Advantages of the Kaplan-Meier estimator include wide applicability, ease of computation, and ease of interpretation. However, the Kaplan-Meier estimator requires information on when censoring occurred, and this information is not readily available in the car-sharing context. For example, in a revenue management context the Kaplan-Meier estimator can be used to estimate total demand; the data required are how many of a product sold each day and whether the sales were right-censored on each day because sales equaled available stock. However, in car-sharing, there are combinatorially many partially substitutable reservations, and it’s not a priori clear when a particular reservation type is “out of stock” in the sense that no acceptable substitute reservation is available.

The revenue management literature also contains machine learning-based nonparametric techniques for unconstraining. Ferreira et al. (2015) use sales data from products which did not sell out to construct nonparametric curve estimates of the portion of total sales as a function of the time a product has been available, and these curves are used to infer total demand for products which did sell out. In car-sharing, we might consider a reservation sold out if that reservation and all acceptable substitutes are taken. Since substitution preferences are not well known in the car-sharing context, the information on which reservations effectively sold out is not readily available.

Our proposed detruncation procedure provides the following methodological contributions. Our method is applicable in cases where the parametric form of the latent distribution is unknown, when the latent distribution may be high-dimensional or have non-real support, and when the available data does not indicate when truncation occurred. Thus our resampling method is appropriate in many contexts beyond car-sharing. We do, however, require knowledge of the truncation procedure, or at least a plausible proxy thereof. This knowledge can be in the form of a simulator; an analytic likelihood is not needed. In addition, our method is highly data-driven: it requires fitting few parameters (e.g., one and two parameters are used in the case studies of Sections 4.2 and 4.3, respectively) and no explanatory models. The principle disadvantages of our method are that: (i) it requires detailed knowledge of the truncation procedure; (ii) it is most applicable when the latent distribution and truncated data have similar support; (iii) for some problems, it is more computationally expensive than other methods, though we note that a single standard laptop has enough computational power to apply our method to the Boston and Manhattan area case studies (Sections 4.2 and 4.3).

Our method is philosophically and intuitively similar to inverse probability weighting (Horvitz and Thompson 1952, Seaman and White 2013) (IPW) because, like IPW, our method assigns a weight to each post-truncation observation. IPW is a technique for performing analyses on data sampled non-uniformly from a population; the classic application is the Horvitz-Thompson estimator for population means (Horvitz and Thompson 1952) given non-uniform sampling from a population. Frequently, IPW is used to correct for missing data: an analysis is to be performed on data from several subjects, and each subject is either a complete case or has some observations missing. Under IPW, the analysis is limited to complete cases, and each case is weighted by the estimated inverse probability of it being complete; this probability must be modeled using additional information about each case. Classic applications of IPW arise in medical research. For example, some patients in a study may not complete every scheduled office visit, in which case patients with full data could be weighted by the inverse probability of having completed every
office visit. In this sense, IPW is commonly compared to imputation as a technique for dealing with missing data. For a recent survey of inverse probability weighting methods, see Seaman and White (2013). Recent medical research papers which use inverse probability weighting include Kahn et al. (2014) and Ayerbe et al. (2013) wherein IPW is used respectively to validate an unweighted analysis and to correct for missing data. Despite the similarity of our method to IPW, we note two key differences. First, our method does not require modeling the probability of an observation being observed or a case being complete; instead, weights are estimated by solving an optimization problem. Second, the weights in our method are not generally interpretable as inverse probabilities. This is because our method for assigning weights to observations (see Section 2) requires solving an optimization problem, and the optimization problem may have several distinct optima. Some of these distinct optimal weight assignments may not correspond to inverse probability weights. In fact, our method can be used when defining and modeling probabilities is impractical.

Within the car-sharing literature, researchers have approached the problem of understanding demand from several angles. Most often, usage data for an existing car-sharing system are used to calibrate a model of system utilization or user utility, and the distinction between observed demand and unobserved lost demand is not considered. In Stillwater et al. (2009), Stillwater et al. use linear regression to relate monthly hours of usage at each station to explanatory variables such as proximity to public transportation and population density. Here, station utilization is used as a proxy for car-sharing demand. In Kumar and Bierlaire (2012), a similar regression technique is used to inform the location of new car-sharing stations. Stillwater et al. note that using observed utilization as a proxy for demand relies on the strong assumption that the CSO has accurately identified and effectively served all demand for car-sharing services; otherwise the utilization will not accurately represent the demand. However, in practice, these conditions are unlikely to hold. For example, de Lorimier and El-Geneidy (2013) reports that high capacity stations have larger catchment areas (the region from which users may come to the station) than small stations. This could well be because many users cannot find a vehicle at their desired station, and their demand spills over to more distant stations (i.e., a user wants a reservation at one station but ends up making a reservation at another “catching” station); large stations are disproportionately likely to catch spillover demand. This suggests a need for research which investigates the spatial distribution of car-sharing demand while modeling the interaction between available capacity and realized demand.

At least one other study (Ciari et al. 2013) has attempted to address these shortcomings by using a simulation methodology to model car-sharing demand without relying on observed usage as a proxy for demand. Here Ciari et al. add car-sharing as a modal option to MATSim, an agent-based transportation simulator (Horni et al. 2016, Ch. 1); this provides highly granular estimates of the spatial distribution of demand for car-sharing services. Though the initial model does not include car-sharing capacity constraints, Ciari et al. find that the model reproduces empirically observed car-sharing behaviors in Zurich, and a follow-up case study in Berlin (Ciari et al. 2014) improves the model by adding capacity constraints and free-floating car-sharing.

Additionally, researchers within the broader mobility field have considered the demand for on-demand mobility services. Notably, O’Mahony (2015) develops a discrete-event simulator for a station-based bike-sharing mobility system in New York City. Time-varying demands for travel between each pair of stations are estimated from data with corrections
for factors such as truncation due to lack of available bikes at the start of a trip, broken bikes, and bad weather days. The simulator and demand estimation technique are extended in Jian et al. (2016), which introduces gradient-like heuristics for finding strategic assignments of bikes and docks to stations. There, demand for travel between each pair of stations is modeled as a time-varying Poisson process with piecewise-constant rate. The approach of modeling demands as time-varying Poisson processes with rates estimated from data is unlikely to transfer well to the car-sharing setting for two reasons. Firstly, car-sharing stations typically have few vehicles and few reservations per day; thus truncation is common and the amount of historical data is limited. Secondly, in car-sharing there is significant heterogeneity in reservation length; reservations may have durations on the order of hours or weeks. This is in contrast to the bike-sharing case, where bikes are typically ridden from one station to another and reservation duration can be well modeled using travel times.

Our work studies the distribution of car sharing demand using a simpler simulation tool than a complex agent-based simulator such as MATSim. We follow Ciari et al. (2013) in that we use a stochastic simulation methodology to study the spatial distribution of car sharing demand, but we do not use a full transportation simulator. Rather, we simulate a simple customer choice model informed by input from industry stakeholders; as a result, our model requires no input information besides the CSO’s historical reservation data. Our simulator is designed to allow us to simulate the truncation process by which latent car sharing demand becomes observed reservations, and by applying the detruncation procedure in Section 2, we approximately infer the latent demand distribution. As a result, our methodology somewhat differs in focus from Ciari et al.: rather than modeling granular demand at the agent level, we focus on capturing the key relationships between station capacity, the spatial distribution of demand, and the spatial distribution of reservations. In particular, our primary contributions in the area of car-sharing are (i) the proposal and validation of a simulation-based method for approximately recovering a truncated latent distribution given truncated data and the truncation procedure; (ii) the formulation of a novel data-driven car-sharing simulator which uses this method to enable the approximate inference of the spatial distribution of car-sharing demand while making few parametric assumptions; (iii) the application of these methods for large-scale case studies in the Boston and New York City (United States) metropolitan areas using high-resolution reservation data from the main U.S. car-sharing operator that show how this model can provide insights of practical use.

The remainder of this paper formulates the proposed methodology (Section 2) along with validation experiments (Section 3), and its use for Boston and New York City case studies (Section 4). The appendices include notation reference (Appendix A), notes on the Wasserstein distance (Appendix B), additional results from the validation experiments in Section 3 (Appendix C), details on the case study inspired validation experiment (Appendix D), and details on our car-sharing simulator (Appendix E).

2 Formulation of the Detruncation Method

2.1 Formulation of the Detruncation Problem

Let \( D \) be an unknown distribution over a metric space \((S, \zeta)\). Let \( f : \bigcup_{n=1}^{\infty} S^n \to \bigcup_{m=0}^{\infty} S^m \). The function \( f \) is the truncation function, and it is assumed to be known. In practice,
f is typically unknown and approximated by a model \( \hat{f} \), but because the problem of approximating \( f \) is separate from that of recovering \( D \), we simplify the formulation here by assuming \( f \) is known.

Unobserved data \( \mathbf{x} = (x_1, \ldots, x_n) \) are generated iid (independently and identically distributed) from \( D \). We see data \( \mathbf{y} = (y_1, \ldots, y_m) = f(x_1, \ldots, x_n) \). Given data \( \mathbf{y} \) and the truncation function \( f \), our goal is to recover the unknown distribution \( D \). In practice, the unavailability of \( D \) prevents us from knowing how close an inferred \( \hat{D} \) is to \( D \), but in synthetic experiments where \( D \) is known, any metric between distributions can be used.

To build intuition, we give an example of what these variables could mean in a car-sharing context. \( S \) could be the space of all possible desired round trip car-sharing reservations with \( \zeta \) the dissimilarity between partially substitutable reservations, and \( D \) could be the probability distribution from which desired reservations are drawn. Then unobserved data \( \mathbf{x} \) would be the set of reservations that users desire. Truncation function \( f \) would be the process of users attempting to make their desired reservations, sometimes having to take a substitute reservation or no reservation at all if the desired reservation is unavailable. Finally, observed data \( \mathbf{y} \) would be the set of reservations actually made by users and thus observed by the CSO.

### 2.2 Formulation of the Resampling Method

We define a weight function \( w : S \to \mathbb{R}_{\geq 0} \) and let \( \hat{D} \) be the discrete distribution with support \( \mathbf{y} \) and probability mass function \( \mathbb{P} \left[ \hat{D} = y_i \right] = \frac{w(y_i)}{\sum_j w(y_j)} \) for \( i = 1, \ldots, m \).

We abbreviate this relationship as \( \hat{D} = w(\mathbf{y}) \). Intuitively, we want to pick a weight function \( w \) such that when we sample iid from \( \hat{D} = w(\mathbf{y}) \) and apply \( f \) to this sample, we get a result \( \hat{y} \) close to observed data \( \mathbf{y} \). This is formalized by the following problem:

\[
\begin{align*}
\text{minimize} & \quad \mathbb{E}[W(\mathbf{y}, \hat{y})] \quad & (1a) \\
\text{subject to} & \quad \hat{D} = w(\mathbf{y}) \quad & (1b) \\
& \quad (z_1, \ldots, z_\eta) \overset{iid}{\sim} \hat{D} \quad & (1c) \\
& \quad \hat{y} = f(z_1, \ldots, z_\eta) \quad & (1d) \\
& \quad w(y_i) \geq 0 \quad \text{for } i = 1, \ldots, m. & (1e)
\end{align*}
\]

Here \( \eta \) is a positive integer denoting how many observations to draw iid from \( \hat{D} \) before applying \( f \), the \( z_i \) are drawn from \( \hat{D} \), and \( W \) is the Wasserstein distance. In this formulation, constraints (1b)–(1d) link observed data \( \mathbf{y} \), estimated latent distribution \( \hat{D} \), and simulated resampled-truncated data \( \hat{y} \). That is, \( \hat{D} \) is a weighted empirical sample of \( \mathbf{y} \), as described above. From \( \hat{D} \) we sample \( \mathbf{z} \) with each \( z_i \) independent, intuitively representing the simulated unseen data \( \mathbf{x} \). Applying \( f \) to \( \mathbf{z} \) yields \( \hat{y} \), the simulated resampled-truncated data, which should be close to the observed data \( \mathbf{y} \). The objective function (1a) is the expected distance between observed data \( \mathbf{y} \) and simulated data \( \hat{y} \); the expectation is over any randomness inherent in the truncation function \( f \) and is typically approximated by sample averaging. We use the Wasserstein distance \( W(\cdot, \cdot) \) as our measure of closeness. The Wasserstein distance (also known as the earth mover’s distance) has a centuries-long history; for a recent paper motivating and defining the distance, see Rubner et al. (2000, Sec. 4). Appendix B contains some definitions and computational details for the Wasserstein distance.
We use the Wasserstein distance rather than other measures of dissimilarity between distributions because it encodes the underlying metric space’s notion of similarity. That is, the Wasserstein distance between two distributions over a metric space \((S, \zeta)\) depends not just on the relative probability mass the distributions assign to outcomes \(s \in S\) but also the distance \(\zeta(s, s')\) between outcomes. To illustrate: in our car-sharing context, \(D\) is the unknown distribution of desired car-sharing reservations. Since car-sharing reservations are partially exchangeable goods, it is desirable for a reservation \(x\) drawn from \(\hat{D}\) to be close to a plausible reservation \(x'\) from \(D\), even if \(x\) itself could not have been generated by \(D\). For example, suppose \(D\) assigns large probability mass to 1-hour reservations. It is better for \(\hat{D}\) to move this mass to 1.5-hour reservations than 15-hour reservations exactly because 1.5-hour reservations are “closer” to 1-hour reservations than 15-hour reservations are.

In Problem 1, the weights \(w\) can be any non-negative function of the observed data \(y\). Since optimizing over all non-negative functions is generally intractable, in practice \(w\) is parameterized. The exact parameterization can depend on the space \((S, \zeta)\). For example, if \((S, \zeta)\) is the reals with the usual absolute value metric, we commonly let \(w\) be a \(k\)-degree polynomial:

\[
\min_{\alpha_1, \ldots, \alpha_k} \mathbb{E}[W(y, \hat{y})] \tag{2a}
\]

subject to

\[
\hat{D} = w(y) \tag{2b}
\]

\[
(z_1, \ldots, z_\eta) \overset{iid}{\sim} \hat{D} \tag{2c}
\]

\[
\hat{y} = f(z_1, \ldots, z_\eta) \tag{2d}
\]

\[
w(y_i) = 1 + \sum_{j=1}^{k} \alpha_j \cdot y_i^j \text{ for } i = 1, \ldots, m \tag{2e}
\]

\[
w(y_i) \geq 0 \text{ for } i = 1, \ldots, m \tag{2f}
\]

\[
\alpha_j \in \mathbb{R} \text{ for } i = 1, \ldots, k. \tag{2g}
\]

In practice, the last constraint that \(w(y_i) \geq 0 \forall i\) often means many parameter values \(\alpha\) will be infeasible because they would lead to \(w(y_i) < 0\) for at least one \(i\). We often find it convenient to slightly alter the formulation so that almost all parameter values will be feasible. Under the reformulation, negative weights are replaced with zero, and a parameter vector \(\alpha\) is feasible if at least one weight is strictly positive. This reformulation expands the set of feasible weights \(w\) and thus should also lead to an improved optimal
The formulation in Problem 3 is used throughout the validation experiments in this paper. The principal assumption of Problem 3 is that the weights $w$ are a polynomial function of the observed data. The resulting formulation is typically quite tractable, especially for a low-dimensional polynomial.

### 2.3 Strengths and Weaknesses of the Proposed Method

Our resampling method is highly data-driven, and this introduces several advantages and disadvantages. One key advantage is that our method does not require specifying a functional form of the latent distribution $\mathcal{D}$; such prior information may not be readily available. Instead, the method assumes $\mathcal{D}$ is a discrete distribution with support $\mathbf{y}$. This is a strong assumption; intuitively, it means the resampling method should only be applied when the random variable $f(\mathcal{D})$ has approximately the same support as $\mathcal{D}$.

In the car-sharing context, this is the case: any reservation $y_i$ that was successfully made represents a reservation that someone actually wanted, but the data $\mathbf{y}$ likely overstate the relative frequency of some reservations such as short reservations, reservations at off-peak times, etc. However, it’s easy to imagine situations where $f$ meaningfully changes the support. For example, suppose $\mathcal{D}$ is the univariate uniform distribution over the interval $[5, 6]$ and $f(x_1, \ldots, x_n) = (x_1 - 5, \ldots, x_n - 5)$. Then the data $\mathbf{y}$ are iid draws from the univariate uniform distribution over $[0, 1]$, and no weighting scheme applied to $\mathbf{y}$ can recover $\mathcal{D}$. If we assume that $\mathcal{D}$ is a distribution over a normed vector space, rather than a simple metric space, the resampling method can be extended by adding a parameterized offset $\psi$ to $\hat{\mathcal{D}}$, i.e., $\hat{\mathcal{D}} = w(\mathbf{y}) + \psi$. This broadens the applicability of the method at the cost of more parameters to fit and greater danger of overfitting. We do not consider this extension in this work.

Applying the resampling method requires solving Problem 3, which is typically a simulation-based optimization problem because function $f$ is often a simulation-based model. In rare cases the truncation function $f$ may be sufficiently simple that a closed form expression for $\hat{\mathbf{y}}$ can be written (and in these cases MLE may be more appropriate), but in general simulation will be required, especially because the method was designed specifically for cases where $f$ is a simulation and thus a likelihood is unavailable. In both our validation experiments (Section 3) and our car-sharing application (Section 4), the weights $w$ are parameterized by just one to two decision variables, so the problem is
low dimensional. Nonetheless, the resampling method is computationally expensive compared to maximum likelihood methods when the likelihood can be written in closed form and directly maximized. For more complicated problems requiring methods such as the expectation-maximization algorithm, we believe the relative efficiency of different methods depends heavily on problem-specific features such as the efficiency of simulating $f$.

3 Validation

We benchmark our resampling detruncation method by comparison to maximum likelihood estimation. MLE is a natural benchmark both because of its ubiquity and its theoretical efficiency. Under mild assumptions the MLE is known to be asymptotically optimal and empirically performs well even on small samples. Thus for benchmarks where the necessary assumptions (the parametric form of $D$ is known and both the truncation function $f$ and the likelihood can be described analytically) hold, the MLE results can be seen as a bound on how well any detruncation method can perform. That is, in such cases no detruncation method can perform asymptotically better than MLE, and the observed practicality of MLE even with limited data should lead us to believe that no detruncation method will meaningfully outperform MLE.

It is important to note that comparisons between our resampling detruncation and maximum likelihood necessarily give MLE an advantage: maximum likelihood estimation can only be performed when an analytic likelihood can be written and optimized, whereas our procedure relies on the weaker assumption that the truncation can be simulated. Additionally, MLE relies on prior knowledge about the parametric form of the latent distribution which may not be available, and our method does not require the practitioner to specify what kind of distribution generated the data. Experiments where the benchmark maximum likelihood estimation is done with correct or nearly-correct knowledge of the form of the latent distribution should be seen as giving MLE the benefit of accurate background knowledge, which our method does not use.

We present two groups of validation experiments. Section 3.1 contains experiments where the underlying metric space is the real line, and we vary features such as the latent distribution, the number of data points observed, the parametric type of the latent distribution used for MLE, etc. This section presents results which suggest that our method is competitive with or even out-performs MLE when the form of the latent distribution is incorrectly guessed. Section 3.2 contains a higher dimensional experiment heavily inspired by our car-sharing case studies in Section 4.

3.1 One Dimensional Experiments

At a high level, our validation experiments are very simple. First, data $x$ are generated iid from latent distribution $D$, which is known in these synthetic experiments. A truncation function $f$, also known perfectly in these experiments, is applied, yielding $f(x) = y$. Given observed data $y$, we create two estimates of $D$: $\hat{D}_{\text{MLE}}$ using maximum likelihood estimation and $\hat{D}$ using our resampling method. We then compare the Wasserstein distances $W(\hat{D}_{\text{MLE}}, D)$ and $W(\hat{D}, D)$ to measure how far each estimate is from the true distribution. The Wasserstein distances $W(\cdot, D)$ are themselves random variables due to randomness in $x, f, y$, and resampled data $z \sim w(y)$. Therefore, for $D' \in \{\hat{D}_{\text{MLE}}, D\}$ we estimate $E[W(D', D)]$ by sample averaging: $\bar{W}(D', D) = \frac{1}{k} \sum_{i=1}^{k} W(D', D)_i$, where each $W(D', D)_i$
is an independent (separate data \(x\) and \(y\), separate inference) realization of \(W(D', D)\). In our experiments, we let \(k = 5\).

The above specification still allows significant freedom in choosing the latent distribution, how many data points \(x\) there are, whether the maximum likelihood estimation problem is correctly specified, and so forth. We allow these experimental parameters to vary independently; that is, we perform an experiment for each unique combination of experimental parameters described below. Each of these elements is now described in more detail.

- **Latent distribution** We let the latent distribution \(D\) be a common continuous univariate distribution. In particular we consider the exponential distribution with scale 2, the normal distribution with unit variance, the Weibull distribution with shape 3 and unit scale, and the log-normal distribution where the underlying normal has expectation 1 and standard deviation 2. These distributions include a variety of qualitative shapes while also having cumulative distribution functions which can be easily differentiated, allowing for efficient MLE inference. We include a location parameter in each distribution’s parameterization, including those distributions (such as the exponential distribution) normally parameterized without a location parameter. In these cases, we use the standard trick \(F_l(x) = F(x - l)\) where \(F\) is the standard location-parameter-free cumulative distribution function, and \(F_l\) is the cumulative distribution function with location parameter \(l\). For each distribution, we consider location parameters in \(\{0, 0.05, 0.1, 0.2\}\).

- **Data size** We define the data size \(n\) as \(n = |x|\), i.e., the number of pre-truncation data points. We consider \(n \in \{10, 20, \ldots, 100, 150, 200, \ldots, 2000\}\).

- **Truncation function** We consider a single truncation function \(f\) which first partitions elements of \(x\) into sets of size two uniformly at random and then keeps the minimum element of each set while truncating the maximum element of each pair. Since the \(x\) are assumed to be iid, we can write this as \(y_i = \min(x_{2i-1}, x_{2i}), i = 1, \ldots, n/2\).

- **Maximum likelihood estimation** For each type of distribution (exponential, normal, Weibull, log-normal) that may have given rise to the pre-truncation data, we perform a separate MLE inference. Since in these synthetic experiments we know the true latent distribution, we can examine cases where the MLE inference is correctly or incorrectly specified. Additionally, the MLE inference is always done assuming a location parameter of 0; maximum likelihood is used to infer only the scale and, as appropriate, shape parameters. Since we consider location parameters \(\{0, 0.05, 0.1, 0.2\}\) for each type of latent distribution, this allows us to examine cases where the MLE inference is not perfectly specified even if the correct type of distribution has been chosen.

- **Resampling estimation** We solve the parameterized simulation-based optimization problem (Problem 3) using \(k = 2\), i.e., the weight \(w(y)\) of an observation \(y\) is a quadratic polynomial in \(y\). We approximately solve this two-dimensional problem using a grid search with resolution 0.1 over the region \([-0.5, 4] \times [-0.5, 4]\). Empirically we find that increasing the resolution of the grid search has little effect on final solution quality. While we don’t generally recommend brute force methods such as grid search, we use grid search in these validation experiments to allay any concerns that the algorithm used to solve Problem 3 does not find a high quality solution.
All together, we consider four families of latent distributions with four location parameter values each, for a total of 16 latent distributions. We consider 48 different data sizes and four assumed distribution families for MLE inference. So the total number of MLE inferences performed is $5 \times 16 \times 48 \times 4 = 15360$, and the total number of resampling inferences performed is $5 \times 16 \times 48 = 3840$. The complete results from these thousands of experiments are too numerous to detail here, but we include some important and representative results below.

Figures 1 through 4 summarize results from cases where the maximum likelihood estimation is performed assuming the correct distributional form of $D$. Each of these figures has four subplots corresponding to the four location parameters we consider. In each subplot, the number of pre-truncation data points $n$ is along the horizontal axis and estimated Wasserstein errors $\bar{W}(D, D')$ for an inferred distribution $D'$ (either $D$ resampling or $D_{\text{MLE}}$) are on the vertical axis. For each value of $n$ we perform 5 trials, where each trial is an independent process of generating data $x$ and $y$ and estimating $D, D_{\text{MLE}}$; the plots show the 5-trial averages. The trend lines are degree-2 local polynomial regression (Cleveland 1979) (LOESS) curves. In all the figures, dotted lines represent our resampling approach, and solid lines represent maximum likelihood.

For example, Figure 1a in the upper-left of Figure 1 corresponds to $D$ being an exponential distribution with location parameter 0 and maximum likelihood inference performed assuming that $D$ is an exponential distribution with location parameter 0, i.e., correctly specified. Figures 1b – 1d display the results for experiments where $D$ is an exponential distribution with location parameter greater than zero. Figure 1 shows that errors of both the MLE and resampling methods decrease at similar rates as the number of data points $n$ increases. Perfectly specified MLE asymptotically approaches 0 error, while the error of the resampling method plateaus higher. However, when $D$ has a nonzero location parameter, the errors of the MLE method are higher than when $D$ has a location parameter of zero because by construction the MLE always assumes a location parameter of zero. When the location parameter is 0.1 (Figure 1c), MLE and resampling are competitive, and when the location parameter is 0.2 (Figure 1d), we see that the resampling approach has better performance than MLE except for very small data sizes.

Figure 3 is organized the same way as Figure 1 but shows results for when $D$ is a Weibull distribution. For all four location parameters considered, the errors of the MLE method are lower than the errors of the resampling method: in each case, by $n = 500$ the errors of the resampling method have plateaued to approximately .08 and the errors of the MLE method have plateaued to approximately .02. Here, we see that the MLE performance does not meaningfully change as the location parameter varies. The Weibull distribution has two parameters, shape and scale, and by varying these jointly, MLE can find a location-zero Weibull which closely matches $D$, even when $D$ has a small but positive location parameter.

Figure 2 shows results for when $D$ is a normal distribution. For all four location parameters considered, the errors of the MLE method are lower than the errors of the resampling method: in each case, by $n = 500$ the errors of the resampling method have plateaued to approximately 0.17, and for all four location parameters, after about 500 data points the errors of the MLE method level off around the location parameter. E.g. when the location parameter is 0.1 (and the MLE is still done with a location parameter of 0), the errors of the MLE method approach 0.1. Therefore we see that the MLE performs better than resampling for location parameters in $\{0, 0.05, 0.1\}$ and worse for a location parameter of 0.2.
Figure 4 shows results for when $D$ is a log-normal distribution. As with the normal distribution, for location parameters in $\{0, 0.05, 0.1\}$ the MLE approach beats resampling, and for a location parameter of 0.2 the resampling approach is competitive with MLE, especially for $n \geq 1000$. The log-normal distribution has a heavy tail and therefore the Wasserstein errors (which involve an integral over the whole real line) are higher for a log-normal distribution $D$ than for the other considered distributions. We also note one feature of interest: for very small $n$, occasionally (e.g., consider $n = 30$ for location parameter 0) inherent randomness in the data leads the maximum likelihood inference to learn a log-scale normal distribution quite far from the true log-scale normal distribution. Errors in the inferred log-scale distribution are magnified through exponentiation, so this leads to quite large errors of the MLE method. For example, consider Figure 4a, which shows results from a perfectly specified MLE procedure. At $n = 30$ there is a spike in the errors of the MLE method arising from a single trial in which MLE performs poorly. Empirically, this problem is less severe with the resampling method; observe that in each subplot of Figure 4, the average error of the resampling method is lower than the average error of the MLE method for at least one $n \in [0, 50]$.

In the experiments summarized in Figures 1–4, the errors of the resampling method tend not to fall beyond some threshold—for example, around 0.2 when $D$ is an exponential distribution (Figure 1) and 1.7 when $D$ is a normal distribution (Figure 2). This is because the weights used for resampling are here constrained to be a quadratic function of the observed data, and the optimal weighting scheme is nonlinear and non-quadratic. In cases with abundant data, the errors of the resampling method can be further reduced by considering more intricate function formulations for the weighting schemes, though this comes at some computational cost. Specifically, finding optimal weights by solving Problem 3 becomes more challenging as the number of decision variables in Problem 3 increases.

We now consider two cases where the MLE inference is imperfectly specified. First, we consider cases where the maximum likelihood inference is performed under distributional misspecification. That is, for a given latent distribution $D$, we perform maximum likelihood inference for each of the distributional types we consider (exponential, Weibull, normal, log-normal) except the type that actually matches $D$, and we report the best-performing of these misspecified estimates. For example, if $D$ is an exponential distribution, we take the best-performing non-Weibull maximum likelihood estimator assuming that $D$ is Weibull, normal, or log-normal. In all these trials, we let $n = 100$ and set location parameters to 0, which matches the maximum likelihood assumption. Table 1 summarizes these results. Each row of the table corresponds to a given $D$. The columns respectively indicate the parametric form of $D$, the best performing (among the misspecified) MLE inference, the average error of the MLE method, and the average error of the resampling method. For example, when $D$ is Weibull, the best-performing non-Weibull maximum likelihood estimator is the log-normal estimator with a Wasserstein loss of $W(D, \hat{D}_{MLE}) = 0.13$. Overall, Table 1 shows that the Wasserstein errors $W(D, \cdot)$ of maximum likelihood with distributional misspecification may be slightly larger or smaller than the corresponding errors of the resampling method. The case of when $D$ is an exponential distribution (first row of the table) is particularly interesting: the Weibull distribution generalizes the exponential distribution, so maximum likelihood inference assuming a Weibull distribution should be able to exactly recover $D$ when $D$ is an exponential distribution. But with only 100 raw (i.e., 50 observed) data points, the flexibility provided by the shape parameter of the Weibull distribution (which

13
Figure 1: Wasserstein errors of the resampling and MLE methods when $D$ is exponentially distributed

Figure 2: Wasserstein errors of the resampling and MLE methods when $D$ is normally distributed
Figure 3: Wasserstein errors of the resampling and MLE methods when $D$ is a Weibull distribution

Figure 4: Wasserstein errors of the resampling and MLE methods when $D$ is log-normally distributed
should be exactly 1 for the Weibull to reduce to an exponential) allows over-fitting, and the errors of the MLE and resampling methods have similar orders of magnitude.

We also consider parametric (rather than distributional) misspecification by giving $D$ a nonzero location parameter; recall that the MLE inference assumes a location parameter of zero. In this case, we report the best MLE estimate, which may be from assuming the correct distribution type or another type. We again consider those trials with $n = 100$ and location parameters of 0.05, 0.1, and 0.2. These results are summarized in Tables 2 through 4. Table 2 shows results from the experiments where the latent distributions $D$ have a location parameter of 0.05; since the MLE is always performed assuming a location parameter of 0, the MLE is nearly perfectly specified. In these cases, the MLE always outperforms our resampling strategy, roughly by a factor of 3 (though except in the case of $D_a$ a log-normal distribution, absolute differences in errors are small). Similarly, Table 3 shows results from experiments where the latent distributions have a location parameter equal to 0.1, i.e., the MLE misspecification is more severe. Here the MLE again outperforms the resampling approach, typically by a factor of about 2. Table 4 shows results from experiments where the latent distributions have a location parameter set to 0.2, and the MLE misspecification is worse still. Here, the MLE outperforms our resampling approach in only three out of four cases, and for those cases where the MLE outperforms our resampling approach, it does so by a factor of about 1.5. In general, we observe that the MLE performance degrades meaningfully as the misspecification of the location parameter grows, while (subject to randomness in the data) the resampling performance does not. This further emphasizes the appropriateness of the resampling method when little distributional information is available.

<table>
<thead>
<tr>
<th>$D$</th>
<th>Best MLE</th>
<th>$W(D, \hat{D}_{\text{MLE}})$</th>
<th>$W(D, \hat{D})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>exponential</td>
<td>Weibull</td>
<td>0.28</td>
<td>0.52</td>
</tr>
<tr>
<td>Weibull</td>
<td>log-normal</td>
<td>0.13</td>
<td>0.08</td>
</tr>
<tr>
<td>normal</td>
<td>exponential</td>
<td>1.00</td>
<td>0.24</td>
</tr>
<tr>
<td>log-normal</td>
<td>Weibull</td>
<td>12.86</td>
<td>14.32</td>
</tr>
</tbody>
</table>

Table 1: Average Wasserstein errors under distributional misspecification

<table>
<thead>
<tr>
<th>$D$</th>
<th>Best MLE</th>
<th>$W(D, \hat{D}_{\text{MLE}})$</th>
<th>$W(D, \hat{D})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exponential</td>
<td>Exponential</td>
<td>0.20</td>
<td>0.57</td>
</tr>
<tr>
<td>Weibull</td>
<td>Weibull</td>
<td>0.03</td>
<td>0.09</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal</td>
<td>0.09</td>
<td>0.25</td>
</tr>
<tr>
<td>Log-normal</td>
<td>Log-normal</td>
<td>5.93</td>
<td>14.72</td>
</tr>
</tbody>
</table>

Table 2: Lowest errors of the MLE method under location misspecification of 0.05

<table>
<thead>
<tr>
<th>$D$</th>
<th>Best MLE</th>
<th>$W(D, \hat{D}_{\text{MLE}})$</th>
<th>$W(D, \hat{D})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exponential</td>
<td>Exponential</td>
<td>0.18</td>
<td>0.42</td>
</tr>
<tr>
<td>Weibull</td>
<td>Weibull</td>
<td>0.05</td>
<td>0.07</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal</td>
<td>0.12</td>
<td>0.24</td>
</tr>
<tr>
<td>Log-normal</td>
<td>Log-normal</td>
<td>9.31</td>
<td>17.32</td>
</tr>
</tbody>
</table>

Table 3: Lowest errors of the MLE method under location misspecification of 0.1
Table 4: Lowest errors of the MLE method under location misspecification of 0.2

<table>
<thead>
<tr>
<th>$\mathcal{D}$</th>
<th>Best MLE</th>
<th>$W(\mathcal{D}, \hat{\mathcal{D}}_{\text{MLE}})$</th>
<th>$W(\mathcal{D}, \hat{\mathcal{D}})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exponential</td>
<td>Exponential</td>
<td>0.31</td>
<td>0.52</td>
</tr>
<tr>
<td>Weibull</td>
<td>Weibull</td>
<td>0.04</td>
<td>0.06</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal</td>
<td>0.20</td>
<td>0.18</td>
</tr>
<tr>
<td>Log-normal</td>
<td>Log-normal</td>
<td>10.89</td>
<td>15.30</td>
</tr>
</tbody>
</table>

A larger sample of our results from these validation experiments are available in Appendix C. These results show the same trends highlighted above: our resampling approach is competitive with but not quite as precise as perfectly-specified maximum likelihood inference, but generally is competitive with even slightly incorrectly specified MLE. This reinforces the appropriateness of our resampling approach in cases where MLE is difficult or impractical to apply, e.g. because the distributional form of $\mathcal{D}$ is unknown and cannot be well estimated.

3.2 Case Study Inspired Validation Experiments

In some of our car-sharing work (Section 4 and Fields et al. (2017)), we consider weights on historical reservations of the form

$$w(r) = 1 + \alpha_1 \cdot l(r) + \alpha_2 \cdot q(r),$$

where $l(r)$ is the duration (in hours) of reservation $r$ and $q(r)$ is the number of weekend (noon on Friday through 11:59pm on Sunday) hours covered by reservation $r$. These attributes—duration and weekend hours covered—are computed from the historical reservation data, which do not include these attributes but instead contain low-level reservation attributes such as reservation start time and reservation end time. The aim of this section is to validate our resampling method in a similar context. The key aspects of the situation that we wish to emulate in a synthetic validation experiment are: i) the underlying distribution $\mathcal{D}$ from which data $x$ are drawn is multi-dimensional; ii) for each data point, a one dimensional feature of interest can be computed; iii) observed data $y$ should be weighted by this one dimensional feature so that the average feature value in $y$ matches the average feature value in $f(\hat{\mathcal{D}})$.

We construct a validation experiment to mimic these features as follows.

- Unobserved data $x$ are drawn from $\mathcal{D}$ which is a multivariate normal distribution with parameters $\mu$ and $\Sigma$.

- The truncation function $f$ acts independently on each element of $x$. An element $x_i$ is observed with probability $1/(1 + \|x_i\|_2)$; note the use of the 2-norm as when $\mathcal{D}$ is multidimensional, each $x_i$ is also a vector. More formally, we can characterize the observed data $y$ by $y = \{x_i \in x : u_i \leq 1/(1 + \|x_i\|_2)\}$ where each $u_i$ is independently distributed uniformly over the interval $[0, 1]$.

- Weights $w(y)$ are of the form $w(y_i) = 1 + \alpha \cdot \|y_i\|_2$.

- The likelihood of resampling an observation $x$ is proportional to $\mathbb{P}[x] \cdot \mathbb{P}[x \text{ is observed}|x \text{ generated}] \cdot w(x)$, where $\mathbb{P}[x]$ is the likelihood of drawing $x$ from $\mathcal{D}$. That is, the likelihood of
resampling an observation $x$ is the product of (i) the likelihood of generating $x \sim D$, (ii) the probability $x$ is not truncated but instead is observed, and (iii) the weight $w(x)$. For resampling to reproduce $D$, this likelihood should be proportional to $P[x]$, meaning $w(x) \cdot P[y|x \text{ observed} | x \text{ generated}]$ is proportional to one. Thus the optimal coefficient is $\alpha = 1$ so that for any $x$, $w(x) = 1 + \|x\|^2$, i.e. the reciprocal of the probability $x$ is truncated. With this optimal coefficient, $\hat{D}$ converges to $D$ in distribution as the size of $x$ goes to infinity.

• In the car-sharing case we choose the coefficient on reservation weekend hours so that the simulated fraction of reservation-hours on weekends matches the fraction of historical reservation-hours on weekends. Here we choose $\alpha$ to match the mean norm of the observed data $y$ and the simulated (via resampling and applying the truncation function) data $\hat{y}$ by minimizing

$$\frac{\sum\|y_i\|}{\#(y)} - \frac{\sum\|\hat{y}_i\|}{\#(\hat{y})}$$

where $\hat{y} = f(D)$, i.e., the result of applying the truncation function $f$ to our resampled estimate of the latent distribution $D$, and $\#(\cdot)$ indicates the length of a list of observations.

To run the experiment as described, we must also specify:

• The dimension $k$ of the latent distribution $D$. We consider $k \in \{1, 2, 3, 4\}$.

• The parameters $\mu$ and $\Sigma$ defining $D$ where $\mu$ is a length-$k$ vector and $\Sigma$ is a $k \times k$ positive semi-definite matrix. Note that defining $D$ thusly requires $(3k + k^2)/2$ parameters, so the number of parameters grows quickly in the dimension of the problem. In defining $D$, we let $\mu = 0$ and have $\Sigma = (M + M')^2$ where $M$ is a $k \times k$ matrix with each element generated iid uniformly from the interval $(-2, 2)$.

• The number $n = |x|$ of unobserved data to be generated independently from $D$. We consider $n \in \{100, 200, \ldots, 1500\}$.

We measure the distance between the true latent distribution $D$ and an inferred distribution $D'$ (either $\hat{D}$ or $\hat{D}_{MLE}$) using the 2-Wasserstein distance $W_2(D, D')$; see Appendix B for details.

Figures 5a and 5b, display results for the experiment with respectively a 1-dimensional and a 2-dimensional latent $D$. In both figures, the horizontal axis represents $n$, i.e., the number of pre-truncation data points generated $\sim D$. The vertical axis represents the final error $W_2(D, D')$ between the true latent distribution $D$ and an inferred $D'$ which may be either $\hat{D}$ (resampling) or $\hat{D}_{MLE}$. Note that for each data size $n$, both the MLE and resampling approaches share the same raw and observed data $x, y$ and thus the errors of the resampling and MLE methods are correlated. The solid and dotted curves are LOESS trend lines for respectively the errors of the maximum likelihood and resampling methods. Both figures show the same trend: for very small $n$ (i.e., little observed data), the maximum likelihood approach outperforms the resampling approach; that is, $W(D, \hat{D}_{MLE}) < W(D, \hat{D})$. This is to be expected since $\hat{D}$ is a discrete distribution and both $D$ and $\hat{D}_{MLE}$ are continuous distributions; a discrete distribution with support of only a few dozen points cannot well approximate a continuous distribution with support of the
whole real line. As \( n \) grows, the performance of both detruncation approaches improves (the errors \( W(D, \cdot) \) decrease), and the performance gap between the maximum likelihood and resampling narrows roughly proportionally.

Our experiments show that the resampling approach is competitive with maximum likelihood based detruncation. We observe that the resampling approach quickly learns the correct coefficient \( \alpha \), meaning truncation and weighted resampling are exact inverse operations. Thus we can consider \( \hat{D} \) as an empirical sample from \( D \). In contrast, \( \hat{D}_{MLE} \) is a continuous distribution with parameters different than those of \( D \). Both methods have errors \( W_2(D, D') \) which decrease at similar rates in the number of data points \( n \), and the magnitudes of the errors are similar. For one dimensional \( D \) the errors are comparable, and in two dimensions the errors of the resampling method are slightly larger than the errors of the MLE method. When \( n \) is small the error of the resampling method \( \bar{W}_2(D, D) \) is typically larger than \( \bar{W}_2(D, \hat{D}_{MLE}) \) primarily because \( \hat{D} \) is a discrete empirical approximation of the continuous \( D \), whereas the inferred MLE distribution is continuous. The effect diminishes with more data points and is more significant in two dimensional space than in one-dimensional space: because high-dimensional space is “bigger” than low dimensional space, discrete approximations suffer more in high-dimensional space.

We do not show graphs in this section representing higher-dimensional \( D \) because when \( D \) is of dimension higher than 2, the likelihood optimization often converges to a pathological solution or a local minimum far from the true parameters, leading to errors of the MLE method many orders of magnitude larger than the errors of the resampling methods. See Appendix D for details on this phenomenon.

4 Car-sharing Demand Case Study

This section illustrates how the proposed approach can be used to make demand inferences of practical and operational interest to car-sharing stakeholders. We begin by describing the problem setting at a high level. Section 4.1 details the application of the resampling method to the car-sharing setting, Section 4.2 describes a case study in the Boston metropolitan area, and Section 4.3 describes a case study in Manhattan, New York.

As mentioned in Section 1, there are many cases where a car-sharing operator (CSO) would like to know the spatial distribution of demand for its services, and historical usage
data is not a reliable proxy for demand. This section gives a high level overview of the problem to be solved, our methodology, and how the resampling technique is utilized. Further details on our car-sharing simulator can be found in Appendix E. For full details on the simulator, see Fields et al. (2017). For more details on how our demand estimation technique and car-sharing simulator can be used to inform operational decisions, see Zhou et al. (2018).

Given historical data on stations, vehicles, and reservations made, a CSO would like to estimate the spatio-temporal distribution of demand for its services. To address this demand inference problem, we consider a set of stations over a specified study period. Each station has a given capacity (number of vehicles assigned) which may vary over the course of the study period; e.g. a station $i$ could have two vehicles assigned for the first week of the study period and one vehicle assigned for the remainder. Our goal is to learn the demand $d_i$ (constant for the study period) at each station $i$ where we define the demand at station $i$ as the expected number of reservation-hours wanted per day at station $i$, i.e., the number of reservation-hours that would be made at station $i$ if station $i$ had infinite vehicles. The inferred station-wise demands $\{d_i\}$ are of practical interest to a CSO for several reasons including identifying which stations are most important and informing supply allocation decisions. For an example of using inferred station-wise demands in a resource allocation context, see Zhou et al. (2018).

Our method uses a data-driven simulator (Fields et al. 2017). Intuitively, the simulator can be understood as mapping station-wise demands $\{d_i\}$ to desired reservations and desired reservations to realized reservations. Denote by $F$ the simulator mapping demands to station-wise utilizations, i.e., $F(d) = \hat{u}$. The utilization of a station (over some predefined study period) is the ratio of reservation-hours made at that station to reservation-hours available at that station. Denote by $u$ the historical utilizations for each station. Then the problem of inferring demands is the simulation-based optimization problem:

$$\minimize_d \|u - E[\hat{u}]\|_2$$
subject to  
$$\hat{u} = F(d)$$
$$d \geq 0,$$

where the demands $d$ are station-wise, as described above.

4.1 Sampling Desired Reservations

Our car-sharing model simulates customers attempting to make their desired reservations, and therefore we need a method for generating plausible desired reservations. Ideally, this method is as data-driven as possible. Unfortunately, the historical reservation data represents successful—rather than desired—reservations. For example, suppose that most users who want to reserve a vehicle for a day or more are unable to find a suitable reservation, but most users who want a vehicle for only one hour can find a reservation. Then many desired long reservations will not make it into the historical data, which will underrepresent the frequency of such desired reservations. This subsection describes how we apply our generic resampling-based detruncation method to the problem of generating plausible desired reservations; necessarily, it includes some methodological detail about our car-sharing simulator.

For simulation purposes, a desired reservation consists of a creation time, a start time, a duration, and a station. Suppose that we wish to generate a desired reservation at station
\( \ell \) on day \( v \) which is the \( i \)-th day of the week. Intuitively, we want to pick a historical reservation at or near \( \ell \) and make a “copy” of that reservation starting on day \( v \). Let \( R \) be the set of historical reservations which occurred in the same postal code as \( \ell \) and started on the \( i \)-th day of the week. We choose a reservation \( r \in R \) according to weights \( w(\cdot) \), i.e., each \( r \) is chosen with probability proportional to \( w(r) \). Then the generated reservation has station \( \ell \), start day \( v \), the same start time of day as \( r \), the same duration as \( r \), and the same lag between creation and start times as \( r \). In other words, the generated desired reservation is a copy of \( r \) shifted temporally so that it starts on day \( v \). For more details on this procedure, please see (Fields et al. 2017).

The weights \( w \) are chosen so that when simulating under weights \( w \), the expected (with respect to simulation stochasticity) average (of simulated reservations) duration of successful reservations \( \bar{l}_{\text{sim}} \) matches the average reservation duration of historical reservations \( \bar{l}_{\text{data}} \). For a reservation \( r_i \), let \( l(r_i) \) be the duration of \( r_i \) in hours. Then we choose the weights \( w \) by solving the following simulation-based optimization problem:

\[
\begin{align*}
\text{minimize} & \quad \left( \bar{l}_{\text{data}} - \mathbb{E}[\bar{l}_{\text{sim}}^w] \right)^2 \\
\text{subject to} & \quad w(r_i) = 1 + \alpha \cdot l(r_i) \quad \forall r_i \in R \\
& \quad w(r_i) > 0 \quad \text{for at least one } r_i \in R \\
& \quad \alpha \in \mathbb{R}
\end{align*}
\]

Note how this problem follows the form of Problem 3. For the case study in section 4.2, we optimize over only a single scalar parameter \( \alpha \) because in our context it is beneficial to have a simple weighting scheme: simple weights provide interpretability and potential insights about how and why truncation occurs. Additionally, using just a small number of parameters limits the size of the physically plausible regions of weight-space. For example, in Problem 6, the last constraint implies \( \alpha > -1 \) since all reservations are at least one hour long. Additionally, we can say with confidence that \( \alpha < 1 \) because letting \( \alpha = 1 \) means weighting 24 hour reservations 12.5 times as much as 1 hour reservations. Such a weighting implies that 24 hour reservations are truncated an order of magnitude more than short reservations, which is inconsistent with expert knowledge. So \( \alpha \in (-1, 1) \) and it is computationally feasible to consider enough weighting schemes to densely cover this region. Since the considered weighting schemes densely cover the feasible region, we can be confident in the correctness and uniqueness of the parameters we find, i.e., we did not overlook any solutions superior (in the sense of Problem 6) to the weighting scheme found. We observe that for \( \alpha \in (-1, 1) \) the expected simulation-based reservation duration \( \mathbb{E}[\bar{l}_{\text{sim}}^w] \) increases monotonically in \( \alpha \), which makes it easy to solve Problem 6 by a simple grid search (see Appendix E.4).

In doing so we find that a duration coefficient of \( \alpha = .0016 \) minimizes the objective function. This value of \( \alpha \) is quite small, but in fact that is reassuring: it suggests the historical data are not too far from (have similar support to) idealized data on desired reservations, and thus our resampling method is a viable detruncation technique. And though \( \alpha \) is small, the effects of the weighting are not negligible. With this weighting scheme, 24 hour long reservations are about 4\% more likely to be chosen than they would be without any weighting. Further, if no weighting is done, we find that the average duration of simulated reservations is about 6.5 hours, roughly 15 minutes shorter than the average duration of historical reservations. Fifteen minutes is a significant discrepancy: the modal reservation duration is just one hour. Thus if the simulator has expected reservation
duration 15 minutes too short, then in expectation for every four simulated reservations, one “extra” hour of vehicle time—long enough for a common reservation—will be available. With the proposed weighting scheme, the average duration of simulated reservations is approximately equal to the average duration of historical reservations. (That is, discrepancy between simulated and historical average reservation durations is due to randomness in the simulator and is typically less than one minute. If we average reservation lengths over multiple simulation runs, the discrepancy in average reservation durations shrinks to the order of seconds.)

4.2 Boston Case Study

For our first case study, the data used come from Zipcar, the largest CSO in the Boston, United States metropolitan area. We use the complete set of Boston-area reservation records from October 2013 through September 2015. To infer station-wise demands, we solve Problem 5 for the month of June 2014, taking as the set of stations considered all Zipcar stations in the Boston area (406 stations). For each station $i$, we also compute $d_U^i$, the average number of reservation-hours made per day (as opposed to inferred $d_i$, which is the estimated mean reservation-hours wanted per day). Average reservation-hours made per day is a convenient measure because unlike station-wise utilizations $u_i \in [0, 1]$, counts of reservation-hours made can be summed across stations.

We estimate demands by solving Problem 5. When we solve this optimization problem multiple times, we find that the resulting demand distributions have pairwise Wasserstein distances of $\approx 100$ meters. For comparison, the mean station-to-station distance is about 1400 meters, and the mean distance between a station and its nearest neighbor is approximately 300 meters. In other words, we consistently infer very similar demand distributions; the results below are from one such typical distribution.

Summed across the city, we estimate that total daily demand for Zipcar round trip services in June 2014 exceeded reservation-hours served by roughly 488 reservation-hours, or roughly 1.2 reservation-hours per station. Table 5 summarizes the station-wise discrepancy between estimated demand $d_i$ and average daily served hours $d_U^i$. At 55% of the stations, the estimated demand $d_i$ is higher than the computed $d_U^i$, suggesting that most stations lack sufficient supply (i.e., assigned vehicles) to meet all demand. Table 5 shows that at some stations, the inferred demand is lower than the realized demand; this discrepancy suggests that these stations are capturing significant spillover from neighboring stations and are less attractive than the utilizations would suggest. We estimate that the station which caught the most spillover reservations served an average of 44.1 more reservation-hours per day than were desired there. Overall, we estimate significant heterogeneity in $d_i - d_U^i$ across stations but that the median station has $d_U^i < d_i$, i.e. demand in excess of served reservation-hours.

<table>
<thead>
<tr>
<th>Quantile</th>
<th>0.0</th>
<th>0.1</th>
<th>0.25</th>
<th>0.5</th>
<th>0.75</th>
<th>0.9</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>-44.1</td>
<td>-11.2</td>
<td>-4.3</td>
<td>0.7</td>
<td>5.2</td>
<td>11.5</td>
<td>90.3</td>
</tr>
</tbody>
</table>

Table 5: Summary of station-wise discrepancies between estimated demand $d_i$ and average daily served reservation-hours $d_U^i$ in June 2014

Figure 6 shows a kernel-smoothed heatmap of the discrepancy $d_i - d_U^i$ between inferred demand $d_i$ and met demand $d_U^i$. Red indicates areas where $d_i > d_U^i$, i.e., areas where we
estimate demand for car-sharing exceeded reservations served. In contrast, green areas indicates sufficient supply, \( d_{i}^{U} \geq d_{i} \). The few deep green areas indicate regions where\( d_{i}^{U} > d_{i} \), i.e., we estimate that these regions attracted so much spillover demand that the reservation-hours served there exceed the desired reservation hours in those regions. Note that the boundaries of the study region are colored green; because we infer demand only at stations, supply is almost always inferred to be sufficient near the boundary of the network.

Interestingly, the inferred station-wise demands indicate that areas of downtown Boston (lower half of the map), despite being car-sharing hot-spots, may have received a relative over-allocation of vehicles. Our model also allows us to approximately infer which areas are net demand sources (generating more demand than served) and demand sinks (serving more demand than generated). In simulation, we track both the original desired station for a reservation and the location (if any) at which the user who wanted that reservation actually reserved a vehicle. This allows us to estimate how often reservations spill to or from each station. In Figures 7a and 7b we show, respectively, the estimated density of spilled reservations (desired reservations which could not be served at the desired stations) and caught reservations (realized reservations served at a station other than the desired station) in June of 2014. In these figures, red indicates areas with many spilled (respectively caught) reservations, while green indicates areas with few such reservations.
The densities of spilled and caught reservations are qualitatively similar because both the number of spilled reservations and the number of caught reservations are highly correlated with the number of reservations served and station capacity, but the discrepancies between the densities of spilled and caught reservations suggest areas that attract customers from elsewhere in the city; insights about spillover can be used to inform station placement or fleet assignment as in Zhou et al. (2018).

4.3 Manhattan Case Study

The above Boston case study illustrates how the proposed approach is well suited to car-sharing demand estimation. In this section, we evaluate how generalizable or transferable the approach is to the analysis of another city with different mobility patterns. If the car-sharing model can perform well—with appropriately chosen reservation weights—in another city with different characteristics than Boston, that would demonstrate the general applicability of our resampling method.

We focus on the Zipcar market of Manhattan (New York). In addition to being a large and significant market for Zipcar, Manhattan also differs significantly from Boston in several aspects. Notably, Manhattan has more dense population, points of interest, and public transportation networks than Boston. Intuitively, we might suspect that these differences would lead to Manhattan having relatively less Zipcar use for everyday errands than Boston. Indeed, the historical data show that median and average reservation durations are longer in Manhattan than in Boston, and long duration (at least 8 hours) weekend trips comprise a larger portion of the historical reservations in Manhattan than in Boston. In other words, the distribution of trip purposes differs from that of Boston, and hence the demand distribution also differs.

Just as in our Boston case study, in Manhattan we attempt to learn the spatial distribution of demand for Zipcar round trip car-sharing services. In particular, we consider the months of April, May, and June 2017 as our study period. For each month $m$ we attempt to learn a vector of station-wise demands $d^m$ with elements $d^m_i$ for each station $i$ in Manhattan. For each of the three months, simulating the Manhattan car-sharing network using the inferred demand vector should reproduce the historically observed station-wise
utilizations. As data we use the network information (station locations and capacities) for the study period as well as all historical Manhattan reservations in the first six months of 2016 and 2017—note that the reservation data cover a broader period than the study period so that the reservation data used are sufficiently rich.

Without any model modification whatsoever—just using Manhattan data in place of Boston data—our model does not perfectly reproduce features of the historical data such as the mean reservation length and the fraction of reservation-hours on weekends. This is unsurprising: the method of weighting historical reservations by their duration and the coefficient used were both chosen in the Boston context. Further, the average distance between inferred demand distributions for Manhattan is approximately 200 meters, whereas the corresponding distance in Boston is approximately 100 meters. An intentional feature of our method is the ability to adjust the reservations weights, for example to account for differing truncation patterns arising in different cities. Informed by stakeholders at Zipcar, for the Manhattan case study we weight historical reservations by

\[ w(r) = 1 + \alpha_1 \cdot l(r) + \alpha_2 \cdot q(r) \] (7)

where \( l(r) \) is the duration in hours of reservation \( r \) and \( q(r) \) is the number of weekend hours covered by reservation \( r \).

Let \( \bar{l}_{\text{data}} \) be the historical average reservation duration, \( R \) the set of historical reservations, and \( E[l^{w}_{\text{sim}}] \) the simulation-based expected reservation duration under weights \( w \), with the expectation over randomness in the simulation. We estimate \( E[l^{w}_{\text{sim}}] \) with \( \hat{E}[l^{w}_{\text{sim}}] \), the sample average over several simulations. Likewise \( \bar{g}_{\text{data}} \) is the historical fraction of reservation-hours which occur on weekends, and \( E[g^{w}_{\text{sim}}] \) is the expected simulation-based fraction of reservation-hours which occur on weekends. Then we choose scalar coefficients \( \alpha_1, \alpha_2 \) so as to solve

\[
\text{minimize}_{\alpha_1, \alpha_2} \quad \left( \frac{\bar{l}_{\text{data}} - E[l^{w}_{\text{sim}}]}{\bar{l}_{\text{data}}} \right)^2 + \left( \frac{\bar{g}_{\text{data}} - E[g^{w}_{\text{sim}}]}{\bar{g}_{\text{data}}} \right)^2 \] (8a)

subject to \[ w(r) = \max \left( 0, 1 + \alpha_1 \cdot l(r) + \alpha_2 \cdot q(r) \right) \quad \forall r \in R \] (8b)

\[ \left( \max_{r \in R} w(r) \right) > 0 \] (8c)

Given that this problem is low dimensional (i.e. of dimension two) and can be solved offline, we choose to solve it via a grid search; see Appendix E.4 for the analogous one-dimensional method.

Solving Problem 8 provides reservation weights tuned for the Manhattan context. With these weights (which are simple in that they depend on only two reservation attributes), we find that simulated average reservation length and simulated fraction of reservation-hours on weekends are both within 2% relative error of the historical values. Additionally, the average Wasserstein distance between independent estimates of the Manhattan-wide demand distribution is on the order of 70 meters, actually surpassing the corresponding value in Boston (approximately 100 meters). Further, the Wasserstein distance between the simulated and historical distributions of reservation-hours is approximately 10 meters, i.e. the historical data is reproduced with high fidelity.

We also tested a variety of other weighting schemes. These include weighting reservations based on higher powers of reservation duration and number of weekend hours covered.
by a reservation, a binary indicator for whether a reservation contained weekend hours,
a binary indicator for whether a reservation started on a weekend, the fraction of all his-
torical reservations starting at the same hour-of-week (in [0, 168)) as the reservation to
be weighted, and the empirical cumulative distribution function of reservation lead time
(amount of time between reservation creation and start time) evaluated at the lead time for
the reservation to be weighted. In general, we found that these more detailed and intricate
weighting schemes did little to improve model performance (as measured by consistency
of demand estimates and distance between the simulated and historical reservation distri-
butions) or interpretability. The effectiveness of the simple weighting methodology used
hints at the general usefulness of simple weighting schemes.

Figure 8 displays results from this Manhattan case study. In this figure, red represents
areas of high demand for Zipcar round trip services, and green represents areas of low
demand. Black dots represent stations. The figure illustrates an important advantage our
method: by inferring a separate demand distribution for each month, we gain information
about long-term trends and time-of-year effects without having to explicitly model these
factors. For example, observe that Figure 8b representing May has more red areas than
Figure 8a. This is because total demand for Zipcar services was higher in May than
in April. In particular this difference is greatly driven by high demand for Memorial day
weekend. Overall, the successful application of the simulation model to Manhattan—a city
with different mobility patterns and demand distributions than Boston—demonstrates the
broad applicability of the resampling method for detruncation.

5 Conclusion

In this paper, we introduced a novel detruncation strategy based on resampling observed
data. Our resampling method simulates the truncation mechanism and uses a weighted
empirical sample of the observed data as an estimate of the latent pre-truncation distri-
bution. The weights are chosen so that truncating a weighted sample of the observed data
approximately reproduces the observed data. Two sets of validation experiments that
consider both well specified and misspecified scenarios show how our resampling strategy complements maximum likelihood estimation, the most widespread detruncation method; in particular, our method is especially applicable when little is known about the latent distribution from which data are drawn before truncation but the truncation mechanism itself can be simulated. In contrast, our method is not as effective as traditional approaches such as maximum likelihood inference when the distributional form of the latent distribution is known and the likelihood can be written analytically. Broadly, our experiments suggest that errors of the resampling method are larger than errors of a correctly-specified MLE method, but both methods’ errors decrease similarly as the amount of available data grows.

We apply the technique to infer car-sharing demand with large scale case studies of the Boston metropolitan area and of Manhattan (New York). The application of the proposed resampling technique to such distinct U.S. markets (e.g., distinct travel demand patterns, distinct mobility services available) demonstrates the broad applicability of the proposed technique and the ease of its transferability from one market to the next. In particular, the formulation of distinct weighting functions provides an opportunity to incorporate expert knowledge without requiring experts to specify the type of demand distribution that underlies the data.

The simple resampling method we present suggests many possible extensions. These include (i) adding a translation to each resampled observation $y$ to account for cases where the truncated data $y$ has different support than the pre-truncation data $x$; (ii) heuristics for identifying features of the observed data $y$ which would be useful when constructing weights $w$; and (iii) algorithms which leverage domain-specific knowledge to efficiently find promising weight functions. Of particular interest are other ways of incorporating expert knowledge. In cases where the truncation mechanism can be simulated, domain experts may have some relevant knowledge about how pre-truncation data are generated, and this knowledge could be used to improve the detruncation performance. For example, in the car-sharing context, experts have knowledge about the variety of underlying trip purposes which might induce a user to search for a reservation. In our ongoing work, we consider how this expert knowledge can allow resampling-type techniques to add richness to or correct bias in small historical samples from intricate distributions. In particular, ongoing work extends the proposed methodology to extract information from user search data, accounting for the discrepancies of the data across devices and users.

Acknowledgments

This work was funded in part by the UPS Doctoral Fellowship and the Ford-MIT Alliance project “Car-sharing services: optimal network expansions to integrate automotive with mass-transit systems and to mitigate congestion at the metropolitan scale”. We are grateful to our collaborators Perry MacNeille at Ford and Arvind Kannan at Zipcar for their ideas and support. This project would not have been possible without generous data and logistical support from Zipcar, especially from Lauren Alexander, Aaron Doody, and Stephen Moseley.
Appendices

A Notation Reference

Table 6 summarizes the key notation used in this paper and, for each symbol, gives the section where that symbol is introduced.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(S, \zeta)$</td>
<td>Metric space</td>
<td>2</td>
</tr>
<tr>
<td>$\mathcal{D}$</td>
<td>Latent data-generative distribution</td>
<td>2</td>
</tr>
<tr>
<td>$x$</td>
<td>Unobserved data generated iid $\sim \mathcal{D}$</td>
<td>2</td>
</tr>
<tr>
<td>$f$</td>
<td>Truncation function</td>
<td>2</td>
</tr>
<tr>
<td>$y$</td>
<td>Observed data</td>
<td>2</td>
</tr>
<tr>
<td>$w(\cdot)$</td>
<td>Weights applied to observed data</td>
<td>2</td>
</tr>
<tr>
<td>$W(\cdot, \cdot)$</td>
<td>Wasserstein distance</td>
<td>2</td>
</tr>
<tr>
<td>$\hat{\mathcal{D}}$</td>
<td>Resampling-based estimate of $\mathcal{D}$</td>
<td>2</td>
</tr>
<tr>
<td>$\hat{D}_{MLE}$</td>
<td>MLE estimate of $\mathcal{D}$</td>
<td>3</td>
</tr>
<tr>
<td>$d_i$</td>
<td>Demand at station $i$</td>
<td>4</td>
</tr>
<tr>
<td>$d_i^h$</td>
<td>Historical reservation-hours reserved at station $i$</td>
<td>4</td>
</tr>
<tr>
<td>$F$</td>
<td>Car-sharing simulator</td>
<td>4</td>
</tr>
<tr>
<td>$u_i$</td>
<td>Historical utilization of station $i$</td>
<td>4</td>
</tr>
<tr>
<td>$\hat{u}_i$</td>
<td>Simulated utilization of station $i$</td>
<td>4</td>
</tr>
<tr>
<td>$r$</td>
<td>Car-sharing reservation</td>
<td>4</td>
</tr>
<tr>
<td>$l(r)$</td>
<td>Duration of reservation $r$ (hours)</td>
<td>4</td>
</tr>
<tr>
<td>$q(r)$</td>
<td>Number of weekend hours covered by reservation $r$</td>
<td>4</td>
</tr>
<tr>
<td>$g$</td>
<td>Fraction of reservation-hours on weekends</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 6: Notation used in this paper
B Wasserstein Distance

We use the Wasserstein distance to measure distance between the true and inferred latent distributions in our validation experiments (Section 3). We also use it to assess the similarity of inferred demand distributions (Section 4). This appendix defines the Wasserstein distance and describes the approximations used.

The $p$-Wasserstein distance between two distributions $\mu$ and $\nu$ on a metric space $(S, \zeta)$ is

$$\inf_{\pi \in \Pi} \left( \int_S \zeta(x,y)^p \, d\pi(x,y) \right)^{1/p},$$

where $\Pi$ is the set of all distributions on $S \times S$ with marginals $\mu$ and $\nu$. Villani (2008, Ch. 6) is a classic source with much more information on optimal transport, the Wasserstein distance, and related topics well beyond the scope of this paper.

The 1-Wasserstein distance (also known as the earth mover’s distance) is particularly intuitive: it measures the “work” done to transform one distribution into another. In physics, work is $(\text{force}) \times (\text{displacement})$; analogously we can think of work here as $(\text{probability mass}) \times (\text{displacement})$. We compute the 1-Wasserstein distance between two discrete distributions $\mu$ and $\nu$ over locations indexed by $1, \ldots, N$ by solving the following linear program. For a location $i$, distributions $\mu$ and $\nu$ respectively assign probability mass $\mu_i$ and $\nu_i$. Note that we assume $\sum_{i=1}^N \mu_i = \sum_{i=1}^N \nu_i = 1$. The distance between locations $i$ and $j$ is $\zeta_{ij}$.

$$\begin{align*}
\text{minimize} & \quad \sum_{i,j \in \{1, \ldots, n\}} \zeta_{ij} f_{ij} \\
\text{subject to} & \quad \sum_{j=1}^n f_{ij} = \mu_i \quad \forall i \in \{1, \ldots, N\} \\
& \quad \sum_{i=1}^n f_{ij} = \nu_j \quad \forall j \in \{1, \ldots, N\} \\
& \quad f_{ij} \geq 0 \quad \forall i, j \in \{1, \ldots, N\}
\end{align*}$$

In one dimension, the Wasserstein distances can be efficiently computed using the observation that when transforming a distribution $\mu$ into a distribution $\nu$, for any $u \in (0, 1)$, the $u$-th quantile of probability mass in distribution $\mu$ will map to the $u$-th quantile of probability mass in distribution $\nu$. So for continuous univariate distributions $\mu$ and $\nu$, we can write

$$W_p(\mu, \nu) = \left( \int_{u=0}^1 |F^{-1}_\mu(u) - F^{-1}_\nu(u)|^p \, du \right)^{1/p}$$

where $F_\mu$ and $F_\nu$ are respectively the cumulative distribution functions of $\mu$ and $\nu$. This integral can be approximated by a Riemannian summation, i.e., chopping the unit interval into $n$ sub-intervals. Denote by $W^n_p(\mu, \nu)$ the Riemannian approximation with $n$ sub-intervals. Then $W_p(\mu, \nu) \approx W^n_p(\mu, \nu)$ where

$$W^n_p(\mu, \nu) = \frac{1 - \frac{2}{n}}{n} \sum_{i=1}^n G \left( \frac{1}{n} + \left( i - \frac{1}{2} \right) \cdot \frac{1 - \frac{2}{n}}{n} \right)$$

29
where \( G(u) = |F^{-1}_\mu(u) - F^{-1}_\nu(u)|^p \). Note that the approximation \( W_p^n(\mu, \nu) \) only discretizes the interval \([1/n, 1 - 1/n]\), but as \( n \) grows the discretized interval approaches the full unit interval. This is because when one of \( \mu \) or \( \nu \) has support equal to the entire real line, \( G(0) \) and \( G(1) \) are either undefined or infinite. For pathological distributions without finite low-order moments, it may be necessary for \( W_p^n \) to discretize a smaller interval like \([\frac{1}{n^{1/2}}, 1 - \frac{1}{n^{1/2}}]\). Since our validation experiments only consider distributions with finite moments, this complication does not affect us. We find that using \( n = 5000 \) provides both computational speed and sufficiently accurate approximations.

In higher dimensions, this approximation fails because it is no longer clear which probability mass under distribution \( \mu \) maps to which probability mass under distribution \( \nu \). In general, discretizations become intractable due to the “curse of dimensionality”: intuitively, higher dimensions are too big. In particular, if a distribution is over \( k \) dimensions and we wish to discretize space into cells such that along each dimension there are at least \( n \) cells, then the number of cells is \( O(n^k) \). Therefore, when computing Wasserstein distances between multivariate normal distributions and/or discrete samples for the case study inspired benchmark, a different approach is needed. Firstly, we use the 2-Wasserstein distance (rather than the intuitively simpler 1-Wasserstein) because the 2-Wasserstein distance between multivariate normal distributions and/or discrete samples for the case study inspired benchmark, a different approach is needed. Secondly, the validation experiment in Section 3.2 requires computing Wasserstein distances \( W(D, \hat{D}) \) between the MLE-inferred distribution and the true latent distribution requires only matrix operations over low-dimensional matrices.

Secondly, the validation experiment in Section 3.2 requires computing Wasserstein distances \( W(D, \hat{D}) \) between the latent distribution \( D \), which is multivariate normal, and the resampled estimate \( \hat{D} \), which is discrete. Let \( D_n \) denote the empirical distribution of \( n \) samples iid from \( D \). It is well known that \( D_n \) converges to \( D \) under the Wasserstein metric (e.g. Fournier and Guillin 2015). Then the triangle inequality gives

\[
W_2(D_n, \hat{D}) - W_2(D_n, D) \leq W_2(D, \hat{D}) \leq W_2(D_n, \hat{D}) + W_2(D_n, D)
\]

And since the terms \( W_2(D_n, D) \to 0 \) as \( n \to \infty \), we can accurately approximate the desired distance \( W_2(D, \hat{D}) \) by taking a large sample from \( D \) and computing the Wasserstein distance between this sample and \( \hat{D} \).

### C Additional Validation Results

In this section we present additional results from our main one-dimensional validation experiment (Section 3.1). Each table corresponds to a given latent distribution \( D \). Within each table, columns correspond to the detruncation strategy used—either maximum likelihood (four possible assumed parametric forms of \( D \)) or our resampling approach. Rows correspond to different data sizes \( n \), the number of pre-truncation data points. Each entry is the corresponding average Wasserstein distance \( \bar{W}(\cdot, D) \) between \( D \) and the inferred (by the method corresponding to the column) latent distribution.

We now highlight some key insights from Tables 7–14. All tables show that as \( n \) grows, the errors of both the MLE and resampling methods decrease. The exceptions are cases
where $\mathcal{D}$ has different support than the type of distribution used for MLE. For example, if $\mathcal{D}$ is exponentially distributed with location parameter 0 and scale parameter 2, then the best-approximating normal distribution with location parameter 0 will be the normal distribution with standard deviation 0, i.e. a Dirac delta distribution; the Wasserstein error will be approximately 2 and increasing $n$ will not improve the error. This situation is visible in Tables 9 and 13 as well as the “Normal” column of Tables 7, 8, 10, 11, 12, and 14.

Tables 7 and 11 show Wasserstein errors for the cases where $\mathcal{D}$ is exponentially distributed with location parameters of 0 and 0.1, respectively. Observe that for $n < 1000$, the Weibull MLE errors are larger than the exponential MLE errors even though the Weibull distribution generalizes the exponential distribution; the extra parameter allows overfitting. For $n = 10$ and a location parameter of 0, the resampling method slightly outperforms maximum likelihood using a Weibull distribution, but MLE is better for $n \geq 50$. For all $n$, resampling greatly outperforms MLE using a normal or log-normal distribution.

Tables 8 and 12 show Wasserstein errors for the cases where $\mathcal{D}$ has a Weibull distribution with location parameters of 0 and 0.1, respectively. In both cases MLE using the correct distribution performs extremely well (reaching an error below 0.1 by $n = 50$), but for $n \geq 50$ the resampling method performs as well as or better than any MLE with distributional misspecification.

Tables 9 and 13 show Wasserstein errors for the cases where $\mathcal{D}$ is normally distributed with location parameters of 0 and 0.1, respectively. In both cases, as $n$ grows the error of the MLE method using a normal distribution quickly approaches the location parameter of $\mathcal{D}$, which equals the amount of location misspecification. MLE using any other distribution performs poorly, as the other considered distributions have only positive support. The resampling method has errors approximately 4 times larger than distributionally correct MLE for location parameter 0 (Table 9) and twice as large for location parameter 0.1 (Table 13).

Finally, tables 10 and 14 show Wasserstein errors for the cases where $\mathcal{D}$ is normally distributed with location parameters of 0 and 0.1, respectively. Because $\mathcal{D}$ has a thick right tail, all methods have large errors compared to the other forms of $\mathcal{D}$ considered. For both location parameters 0 and 0.1, MLE with a log-normal distribution is the best performing method; resampling errors are approximately a factor of four larger for location parameter 0 and a factor of two larger for location parameter 0.1.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Exponential</th>
<th>Weibull</th>
<th>Normal</th>
<th>Log-normal</th>
<th>Resample</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.75</td>
<td>1.34</td>
<td>2.0</td>
<td>10.4</td>
<td>1.05</td>
</tr>
<tr>
<td>50</td>
<td>0.34</td>
<td>0.45</td>
<td>2.0</td>
<td>1.87</td>
<td>0.64</td>
</tr>
<tr>
<td>100</td>
<td>0.19</td>
<td>0.28</td>
<td>2.0</td>
<td>2.18</td>
<td>0.52</td>
</tr>
<tr>
<td>500</td>
<td>0.09</td>
<td>0.11</td>
<td>2.0</td>
<td>2.72</td>
<td>0.3</td>
</tr>
<tr>
<td>1000</td>
<td>0.07</td>
<td>0.07</td>
<td>2.0</td>
<td>2.18</td>
<td>0.22</td>
</tr>
</tbody>
</table>

Table 7: Wasserstein errors when $\mathcal{D}$ is exponentially distributed with location parameter 0
<table>
<thead>
<tr>
<th>$n$</th>
<th>Exponential</th>
<th>Weibull</th>
<th>Normal</th>
<th>Log-normal</th>
<th>Resample</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.56</td>
<td>0.18</td>
<td>0.98</td>
<td>0.17</td>
<td>0.24</td>
</tr>
<tr>
<td>50</td>
<td>0.72</td>
<td>0.06</td>
<td>1.08</td>
<td>0.14</td>
<td>0.11</td>
</tr>
<tr>
<td>100</td>
<td>0.74</td>
<td>0.03</td>
<td>1.09</td>
<td>0.13</td>
<td>0.08</td>
</tr>
<tr>
<td>500</td>
<td>0.75</td>
<td>0.03</td>
<td>1.10</td>
<td>0.15</td>
<td>0.06</td>
</tr>
<tr>
<td>1000</td>
<td>0.72</td>
<td>0.01</td>
<td>1.08</td>
<td>0.13</td>
<td>0.07</td>
</tr>
</tbody>
</table>

Table 8: Wasserstein errors when $D$ has a Weibull distribution with location parameter 0

<table>
<thead>
<tr>
<th>$n$</th>
<th>Exponential</th>
<th>Weibull</th>
<th>Normal</th>
<th>Log-normal</th>
<th>Resample</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1.0</td>
<td>1.0</td>
<td>0.13</td>
<td>4.47</td>
<td>0.52</td>
</tr>
<tr>
<td>50</td>
<td>1.0</td>
<td>1.0</td>
<td>0.09</td>
<td>4.47</td>
<td>0.28</td>
</tr>
<tr>
<td>100</td>
<td>1.0</td>
<td>1.0</td>
<td>0.07</td>
<td>4.47</td>
<td>0.24</td>
</tr>
<tr>
<td>500</td>
<td>1.0</td>
<td>1.0</td>
<td>0.01</td>
<td>4.47</td>
<td>0.16</td>
</tr>
<tr>
<td>1000</td>
<td>1.0</td>
<td>1.0</td>
<td>0.03</td>
<td>4.47</td>
<td>0.16</td>
</tr>
</tbody>
</table>

Table 9: Wasserstein errors when $D$ is normally distributed with location parameter 0

<table>
<thead>
<tr>
<th>$n$</th>
<th>Exponential</th>
<th>Weibull</th>
<th>Normal</th>
<th>Log-normal</th>
<th>Resample</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>16.2</td>
<td>16.48</td>
<td>19.2</td>
<td>14.59</td>
<td>17.94</td>
</tr>
<tr>
<td>50</td>
<td>14.76</td>
<td>13.01</td>
<td>19.2</td>
<td>6.24</td>
<td>15.26</td>
</tr>
<tr>
<td>100</td>
<td>15.05</td>
<td>12.86</td>
<td>19.2</td>
<td>6.85</td>
<td>14.32</td>
</tr>
<tr>
<td>500</td>
<td>14.55</td>
<td>11.54</td>
<td>19.78</td>
<td>5.91</td>
<td>12.58</td>
</tr>
<tr>
<td>1000</td>
<td>14.57</td>
<td>11.61</td>
<td>19.2</td>
<td>2.69</td>
<td>10.79</td>
</tr>
</tbody>
</table>

Table 10: Wasserstein errors when $D$ is log-normally distributed with location parameter 0

<table>
<thead>
<tr>
<th>$n$</th>
<th>Exponential</th>
<th>Weibull</th>
<th>Normal</th>
<th>Log-normal</th>
<th>Resample</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.49</td>
<td>0.79</td>
<td>2.11</td>
<td>2.67</td>
<td>0.91</td>
</tr>
<tr>
<td>50</td>
<td>0.43</td>
<td>0.47</td>
<td>2.1</td>
<td>1.08</td>
<td>0.65</td>
</tr>
<tr>
<td>100</td>
<td>0.18</td>
<td>0.23</td>
<td>2.1</td>
<td>0.52</td>
<td>0.42</td>
</tr>
<tr>
<td>500</td>
<td>0.17</td>
<td>0.16</td>
<td>2.1</td>
<td>0.59</td>
<td>0.26</td>
</tr>
<tr>
<td>1000</td>
<td>0.17</td>
<td>0.19</td>
<td>2.1</td>
<td>0.56</td>
<td>0.23</td>
</tr>
</tbody>
</table>

Table 11: Wasserstein errors when $D$ is exponentially distributed with location parameter 0.1

<table>
<thead>
<tr>
<th>$n$</th>
<th>Exponential</th>
<th>Weibull</th>
<th>Normal</th>
<th>Log-normal</th>
<th>Resample</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.86</td>
<td>0.11</td>
<td>1.23</td>
<td>0.13</td>
<td>0.17</td>
</tr>
<tr>
<td>50</td>
<td>0.85</td>
<td>0.02</td>
<td>1.23</td>
<td>0.09</td>
<td>0.09</td>
</tr>
<tr>
<td>100</td>
<td>0.89</td>
<td>0.05</td>
<td>1.25</td>
<td>0.12</td>
<td>0.07</td>
</tr>
<tr>
<td>500</td>
<td>0.88</td>
<td>0.02</td>
<td>1.24</td>
<td>0.08</td>
<td>0.07</td>
</tr>
<tr>
<td>1000</td>
<td>0.87</td>
<td>0.02</td>
<td>1.24</td>
<td>0.09</td>
<td>0.07</td>
</tr>
</tbody>
</table>

Table 12: Wasserstein errors when $D$ has a Weibull distribution with location parameter 0.1
Table 13: Wasserstein errors when $D$ is normally distributed with location parameter 0.1

<table>
<thead>
<tr>
<th>$n$</th>
<th>Exponential</th>
<th>Weibull</th>
<th>Normal</th>
<th>Log-normal</th>
<th>Resample</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>16.01</td>
<td>15.55</td>
<td>19.81</td>
<td>29.25</td>
<td>18.81</td>
</tr>
<tr>
<td>50</td>
<td>14.28</td>
<td>12.59</td>
<td>19.32</td>
<td>6.31</td>
<td>15.25</td>
</tr>
<tr>
<td>100</td>
<td>14.75</td>
<td>12.64</td>
<td>20.52</td>
<td>9.31</td>
<td>17.32</td>
</tr>
<tr>
<td>500</td>
<td>14.11</td>
<td>11.46</td>
<td>19.75</td>
<td>6.6</td>
<td>11.26</td>
</tr>
<tr>
<td>1000</td>
<td>14.25</td>
<td>11.63</td>
<td>20.07</td>
<td>7.17</td>
<td>10.78</td>
</tr>
</tbody>
</table>

Table 14: Wasserstein errors when $D$ is log-normally distributed with location parameter 0.1

<table>
<thead>
<tr>
<th>$n$</th>
<th>Exponential</th>
<th>Weibull</th>
<th>Normal</th>
<th>Log-normal</th>
<th>Resample</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>16.01</td>
<td>15.55</td>
<td>19.81</td>
<td>29.25</td>
<td>18.81</td>
</tr>
<tr>
<td>50</td>
<td>14.28</td>
<td>12.59</td>
<td>19.32</td>
<td>6.31</td>
<td>15.25</td>
</tr>
<tr>
<td>100</td>
<td>14.75</td>
<td>12.64</td>
<td>20.52</td>
<td>9.31</td>
<td>17.32</td>
</tr>
<tr>
<td>500</td>
<td>14.11</td>
<td>11.46</td>
<td>19.75</td>
<td>6.6</td>
<td>11.26</td>
</tr>
<tr>
<td>1000</td>
<td>14.25</td>
<td>11.63</td>
<td>20.07</td>
<td>7.17</td>
<td>10.78</td>
</tr>
</tbody>
</table>

D Multivariate Normal Validation Details

D.1 Inspiration

This section provides brief details on the case study inspiration behind the validation experiment in Section 3.2. In our car-sharing work (Section 4), unseen data $x$ are desired car-sharing reservations, and the observed data $y$ are reservations users actually made. While deciding on a weighting scheme, subject matter experts told us and the data confirmed that the distinction between weekend and weekday reservations is critical. Comparing simulated successful reservations ($\hat{f}(\hat{D})$) with the historical reservation record $y$, we observed that:

- Roughly 40% of historical reservation-hours fell on weekends. (The exact figure varies by period examined, definition of a weekend, and so forth.)
- With all weights $w(y_i) = 1$, i.e., using a uniform sample from all historical reservations as simulator input, roughly 35% of simulated reservation-hours fell on weekends.
- The fraction of simulated reservation-hours falling on weekends can be increased by weighting each historical reservation by the number of weekend hours it covers.
- The number of weekend hours covered by a reservation is not a feature directly in the data but is instead computed from other reservation attributes: start date, start time of day, and duration.

We wished to create a validation experiment where a single feature computed for each unseen piece of data $x$ strongly determines the truncation behavior and whether that $x$ becomes an observed $y$. In the car-sharing context this feature is the number of weekend hours covered by a reservation; in our validation experiment, it is the norm $\|x\|$.
D.2 Implementation Challenges for the MLE

The formulation in Section 3.2 poses a challenge for maximum likelihood inference. Let $X$ and $Y$ be random variables representing respectively a single draw from $D$ and a single observation. The likelihood of observing $Y = y$ as a function of parameters $\theta$ (mean and covariance of the latent $D$) depends on the probability of a draw from $D$ being observed as a function of $\theta$:

$$P_{\theta}[Y = y] = P_{\theta}[X = y| X \text{ seen}] = \frac{P_{\theta}[X = y] / (1 + \|y\|)}{P_{\theta}[X \text{ seen}]}$$  \hspace{1cm} (13)

Where $X \text{ seen}$ is the event that the draw $X \sim D$ is observed rather than truncated. The probability of observation $P_{\theta}[X \text{ seen}]$ is the expectation $E_{\theta}[1/(1 + \|X\|)]$, and we are not aware of a closed form expression for this expectation. The expectation can be approximated by sample averaging over $N$ draws from $D$:

$$E_{\theta}[1/(1 + \|X\|)] \approx \frac{1}{N} \sum_{i=1}^{N} \frac{1}{1 + \|x_i\|} \hspace{0.5cm} x_1, \ldots, x_N \overset{iid}{\sim} \mathcal{N}(\theta).$$  \hspace{1cm} (14)

Because this expectation is not trivially differentiable, estimating this expectation by sample averaging for every value of $\theta$ considered while maximizing the likelihood would prohibit the use of analytic derivatives for the likelihood maximization procedure. If analytic derivatives cannot be used and an approximation such as finite differencing is used instead, the resulting solution may not actually be optimal; additionally, the run time would increase significantly, likely by orders of magnitude.

Instead, we use a property of multivariate normal random variables: for any $\theta$, the multivariate normal $\mathcal{N}(\theta)$ can be expressed as an affine transformation of a zero-mean, identity-covariance multivariate normal: $\mathcal{N}(\theta) = A(\theta) \cdot Z + \mu(\theta)$ where $A(\theta)$ is a real matrix depending on the covariance of $\mathcal{N}(\theta)$ and $\mu(\theta)$ is the mean of $\mathcal{N}(\theta)$. We generate samples $z_1, \ldots, z_{1000} \overset{iid}{\sim} Z$ where $Z$ is a standard multivariate normal. Then for any $\theta$ we approximate

$$E_{\theta}[1/(1 + \|X\|)] \approx \frac{1}{1000} \sum_{i=1}^{1000} \frac{1}{1 + \|A(\theta)z_i + \mu(\theta)\|} \hspace{0.5cm} z_1, \ldots, z_{1000} \overset{iid}{\sim} Z.$$

By pre-generating the randomness in the form of the $z_i$, we are left with an approximation for $E_{\theta}[1/(1 + \|X\|)]$ which is both continuous and differentiable as a function of $\theta$.

We note this implementation difficulty to illustrate how in some cases, our resampling method can be easier to implement and faster to run than a maximum likelihood strategy. In this experiment, the resampling strategy required no problem-specific code or analysis. In contrast, the maximum likelihood strategy requires careful problem analysis and an external nonlinear optimization library (we use Ipopt with the modeling language JuMP (Dunning et al. 2017) and ForwardDiff.jl (Revels et al. 2016) for automatic differentiation). Further, the maximum likelihood approach is more “brittle” than the resampling approach. For example, if the experiment were modified so that the latent distribution $D$ were no longer a multivariate normal, then our technique for approximating $P_{\theta}[X \text{ seen}]$ would no longer apply, and either a new trick or numerically estimated derivatives would be needed. Because the resampling code makes no distributional assumptions whatsoever, no modification would be required for the resampling approach. The relative runtimes of
the two approaches depend on the problem dimension and, importantly, on the distribution specified for MLE. If the latent distribution $D$ is low dimensional, then the MLE approach requires simply solving a low dimensional continuous optimization problem, and runs faster than the resampling approach. In higher dimensions, the MLE approach slows dramatically as the number of parameters to be optimized increases like $k^2$ for $k$-dimensional $D$; in contrast, the resampling approach has almost dimension-independent runtime. (The only dimension-dependent effect is that in higher dimensions, it takes longer to compute norms $\|x_i\|_\cdot$.)

In our experiments with dimensions higher than three, we observe that the likelihood maximization procedure often fails to find the correct local maximum due to numerical problems in the optimization. The optimization library we use (Ipopt) does eventually find a local maximum or saddle point, but typically one far from the true parameters. This leads to errors of the MLE method $W_2(D, \hat{D}_{MLE})$ thousands of times larger than the errors of the resampling method $W_2(D, \hat{D})$. We don’t claim that the likelihood maximization is ill-posed or impossible to solve, simply that it requires more subtlety than the standard toolbox of off-the-shelf solvers, first-order methods with derivatives provided by automatic differentiation, double precision floating point numbers, and so forth. This further illustrates the relative implementation difficulty of maximum likelihood estimation compared to that of our proposed resampling method in this problem.

E Car-sharing Simulator Overview

This section provides some intuition and details about our car-sharing simulator.

E.1 Demands $\{d_i\}$

We use our car-sharing simulation model to learn a set of station-wise demands $\{d_i\}$. Each demand $d_i$ is a single real number representing the scale of demand at station $i$. We need not learn parameters describing how demand varies cyclically over a day or week; instead, these effects are reproduced in simulation via our use of the rich historical data. For example, more reservations are desired at at 3pm than 3am. We do not model this time of day dependency, by sampling from the historical data which contains more 3pm reservations than 3am reservations, we end up recreating this effect naturally. Note that in this work we do not let the demands vary temporally; as we usually consider a study period of a few weeks or a month (and do account for time of day and day of week effects), this is not a major limitation. More precisely, we cannot claim to learn the true demands $d = \{d_i\}$. Instead, we are inferring a set of plausible station-wise demands using data; there may be many reasonable demand distributions which are compatible with the observed data, and we do not claim to recover the unique true distribution.

E.2 Exogenous Data and Calibration

The simulator requires data on historical reservation data, network information, and two calibration parameters. The network information needed is the network topology—the location of each station and the distances between stations—and data specifying, for any given station and time, which vehicles are assigned to that station at that time. The two input parameters describe user preferences regarding reservation substitution. Intuitively, we model users as first considering their preferred reservation, and should that reservation
be unavailable, considering increasingly dissimilar possible substitute reservations. Thus one of these two parameters is a function defining the dissimilarity between two potential reservations given the distances in time and space between those reservations. The other parameter roughly represents user flexibility; more precisely, it is the (space and time invariant) probability that, failing to find a reservation of dissimilarity $\varepsilon$ from their desired reservation, a user will consider reservations at dissimilarity $\varepsilon + 1$ rather than exiting the system without a reservation. These values of these user-choice parameters are chosen in consultation with the subject matter experts at Zipcar, who in turn are informed by comprehensive user surveys Zipcar performs regularly. See Fields et al. (2017) for more details. In addition, the simulator requires the historical reservation record for the stations of interest; for each historical reservation we need the location, start time, end time, and creation time (when a user made the reservation, thus rendering the vehicle unavailable to other users).

**E.3 Simulation**

Once calibrated with the exogenous data described above, the simulator takes as input the station-wise demands $d$; these station-wise demands define the expected number of desired reservations per day at each station. Each station-wise demand $d_i$ is a non-negative real number. As output, the simulator returns the simulated state of the car-sharing network after all desired reservations have been attempted. In particular, for each station, the simulator returns a list of all reservations at that station, and the simulator also returns a list of all desired reservations which could not be made or substituted and thus led to the user exiting the system. Many statistics can be calculated from this returned information, but the station-wise utilizations are of particular importance.

Once the simulator is calibrated and the demands $d$ are specified, simulation proceeds in three steps. First, a set of simulated desired reservations is generated by sampling from the data on historical reservations; see Section 4.1 for details on the reservation generation methodology. The expected number of desired reservations generated for each station $i$ is proportional to the demand $d_i$ at that station. Second, reservations are simulated in order of their simulated creation times. For each reservation, we simulate a user attempting to reserve a vehicle at the desired time and location. If the desired reservation is available the user will take it; otherwise, an acceptable replacement or no reservation may be chosen. Finally, for each station $i$ we calculate the simulated utilization $\hat{u}_i$, that is, the proportion of available reservation-hours at station $i$ reserved in simulation.

**E.4 Grid Search for Reservation Duration Coefficient $\alpha$**

To find the duration coefficient $\alpha$ for reservation weights (Section 4.1), we use a simple curve-fitting methodology as follows. Let $\lambda(\alpha)$ be the expected Boston-wide average simulated reservation duration when reservations are sampled with weights $w(r_i) = 1 + \alpha \cdot l(r_i)$ (see Problem 6). We require that at least one reservation has non-negative weight, so the smallest $\alpha$ we consider is $\alpha_{\text{min}} = -1$ since the shortest reservations are one hour long. We set $\alpha_{\text{max}} = -\alpha_{\text{min}}$. Let $X = \{\alpha_{\text{min}} + i/20 \cdot (\alpha_{\text{max}} - \alpha_{\text{min}}) | i \in \{0, 1, \ldots, 20\}\}$; this is the discrete set of $\alpha$ values used for curve fitting.

We simulate the city of Boston with duration coefficient $\alpha$ for each $\alpha \in X$, yielding corresponding estimates $\{\hat{\lambda}(\alpha) : \alpha \in X\}$. A degree-2 LOESS curve is fit through the points $(\alpha, \hat{\lambda}(\alpha))$ for $\alpha \in X$; denote the value of this curve at a value $\alpha$ as $\Lambda(x)$. 

36
We always observe $\Lambda$ to be strictly increasing in $\alpha$, which makes sense given that larger $\alpha$ leads to sampling more long reservations. This leads to the intuitive notion of choosing the duration coefficient $\alpha^*$ such that $\Lambda(\alpha^*) = \bar{l}_{\text{data}}$ where $\bar{l}_{\text{data}}$ is the historical average reservation duration. Formally, we choose as the best coefficient $\alpha^* = \inf\{\alpha \in [\alpha_{\text{min}}, \alpha_{\text{max}}] | g(\alpha) \geq \bar{l}_{\text{data}}\}$. That is, intuitively we let $\alpha$ be the smallest value that leads to a simulated average reservation duration at least as long as the historical average reservation duration. The preference for small $\alpha$ is because, all else equal, we prefer weights close to 1 so that the resampled reservations are as similar to the historical data as possible while still reproducing important attributes (such as average reservation length) in simulation.

References


37


