OTClean: Data Cleaning for Conditional Independence Violations using Optimal Transport

ALIREZA PIRHADI, University of Western Ontario, Canada
MOHAMMAD HOSSEIN MOSLEMI, University of Western Ontario, Canada
ALEXANDER CLONINGER, University of California San Diego, USA
MOSTAFA MILANI, University of Western Ontario, Canada
BABAK SALIMI, University of California San Diego, USA

Ensuring Conditional Independence (CI) constraints is pivotal for the development of fair and trustworthy machine learning models. In this paper, we introduce OTClean, a framework that harnesses optimal transport theory for data repair under CI constraints. Optimal transport theory provides a rigorous framework for measuring the discrepancy between probability distributions, thereby ensuring control over data utility. We formulate the data repair problem concerning CIs as a Quadratically Constrained Linear Program (QCLP) and propose an alternating method for its solution. However, this approach faces scalability issues due to the computational cost associated with computing optimal transport distances, such as the Wasserstein distance. To overcome these scalability challenges, we reframe our problem as a regularized optimization problem, enabling us to develop an iterative algorithm inspired by Sinkhorn's matrix scaling algorithm, which efficiently addresses high-dimensional and large-scale data. Through extensive experiments, we demonstrate the efficacy and efficiency of our proposed methods, showcasing their practical utility in real-world data cleaning and preprocessing tasks. Furthermore, we provide comparisons with traditional approaches, highlighting the superiority of our techniques in terms of preserving data utility while ensuring adherence to the desired CI constraints.

CCS Concepts: • **Computing methodologies** → **Machine learning**; • **Information systems** → **Data management systems**.

ACM Reference Format:
Alireza Pirhadi, Mohammad Hossein Moslemi, Alexander Cloninger, Mostafa Milani, and Babak Salimi. 2024. OTClean: Data Cleaning for Conditional Independence Violations using Optimal Transport. *Proc. ACM Manag. Data* 2, 3 (SIGMOD), Article 160 (June 2024), 26 pages. https://doi.org/10.1145/3654963

1 INTRODUCTION

Conditional Independence (CI) plays a pivotal role in probability and statistics. At its core, a CI statement, represented as $(X \perp\!\!\!\perp Y \mid Z)$, implies that when Z is known, the knowledge of X doesn't provide any further insight into Y, and vice versa. To illustrate, consider rainfall (Z) influencing both the wetness of grass (X) and the decision to use an umbrella (Y). If we're already aware that it rained, then determining that the grass is wet doesn't shed any additional light on a person's choice to carry an umbrella. CI is foundational in numerous areas. It underpins causal reasoning and graphical models, serving as a cornerstone for efficient probabilistic inference [34, 41]. In the

Authors' addresses: Alireza Pirhadi, University of Western Ontario, London, Canada, apirhadi@uwo.ca; Mohammad Hossein Moslemi, University of Western Ontario, London, Canada, mmoslem3@uwo.ca; Alexander Cloninger, University of California San Diego, San Diego, CA, USA, acloninger@ucsd.edu; Mostafa Milani, University of Western Ontario, London, Canada, mostafa.milani@uwo.ca; Babak Salimi, University of California San Diego, San Diego, CA, USA, bsalimi@ucsd.edu.

ACM 2836-6573/2024/6-ART160
https://doi.org/10.1145/3654963

realm of machine learning (ML), CI's significance spans across feature selection [35], algorithmic fairness [13, 27, 29, 45–47, 51], representation learning [43], model interpretability [6, 24, 26], transfer learning [44], and domain adaptation [39].

CIs in statistics can be analogized with integrity constraints in databases [55]. Specifically, in the context of databases, dependencies such as Functional Dependencies (FDs), Conditional Functional Dependencies (CFDs), and Multivalued Dependencies (MVDs) encapsulate critical semantic and structural constraints. These constraints are imperative for maintaining data integrity in relational databases and play a pivotal role in tasks like data quality management and data cleaning [8, 10, 20]. In a parallel vein, CI represents key statistical constraints that are indispensable for ensuring the robustness and validity of datasets in domains like ML and statistical inference. To elucidate this analogy further, consider the following example.

Example 1.1. In this example, we underscore the significance of maintaining and enforcing CI constraints in data pipelines as essential steps in constructing fair and and reliable ML models, illustrated within the contexts of medical diagnosis and job applications.

Medical Diagnosis. *Consider a dataset used for predicting patient recovery from respiratory infections, consists of attributes such as patient demographics, including their ZIP code, health measurements, the bacterial strain causing the infection, the prescribed antibiotic, and the recovery outcome. Based on domain knowledge, one would expect that the patient's ZIP code should be independent of the recovery outcome given all causal factors that affect the patient's recovery, i.e.,* ($\textit{ZIP code} \perp\!\!\!\perp \textit{Recovery} \mid \textit{Causal factors}$)*. However, existing biases, such as certain ZIP codes having better healthcare access or particular residents' health behaviors, can introduce spurious associations. Additionally, data quality issues, including incorrect ZIP code entries or inaccurately recorded recovery outcomes, or even systematic data quality issues on other attributes that are distributed non-randomly for patients with different ZIP codes, can also violate this expected independence. Training a model on this dataset may lead to a model that picks up spurious correlations between recovery outcomes and ZIP codes rather than the actual causal factors, affecting the model's performance during deployment. Furthermore, simply dropping ZIP code and not using it for training ML models does not resolve the issue if the constraint is violated due to data quality issues on the selected features. In that case, the performance of the model during deployment becomes different for different subpopulations with different ZIP codes, leading to potential geographic biases.*

Job Application. *Consider a dataset used for making hiring decisions. This dataset consists of attributes from applicants' CVs and insights from interviews, encompassing variables such as hobby, hometown, previous companies worked at, project experiences, and other qualifications. In an ideal scenario, factors considered extraneous, like hobby, university attended, and hometown, should be independent of the hiring decision when conditioned on the applicant's qualifications, i.e.,* ($\textit{Extraneous Factors} \perp\!\!\!\perp \textit{Hiring Decision} \mid \textit{Qualifications}$) *However, this constraint can be violated in the dataset due to various reasons. Biases may emerge if, for example, a significant proportion of successful candidates in the dataset share hobbies perceived as technical or come from specific renowned hometowns. Data quality issues, such as inconsistent categorization of qualifications or historical biases in hiring practices, further compound the issue. These extraneous factors not only divert the model's focus from genuine qualifications but can also inadvertently introduce biases. When these factors correlate with sensitive attributes, such as race and gender, the resulting model may become profoundly unfair.*

In this paper, **we address the problem of repairing a dataset with respect to CI constraints.** Given a dataset that violates a CI constraint due to data biases and data quality issues, our goal is to clean the data to ensure adherence to CI constraints while preserving data utility. Much research has been dedicated to computing optimal repairs for data dependencies, particularly functional

dependencies and conditional functional dependencies [10, 33, 37, 38]. However, the challenge of repairs concerning CI remains relatively unexplored. A significant contribution in this area is the work by Salimi et al. [47]. Their study links CI to Multi-valued dependencies (MVDs) and provides methods to compute optimal repairs by **minimizing the number of tuple deletion and insertion** to ensure consistency with an MVD [47].

A significant challenge in data cleaning for ML is how to ensure that these operations do not distort the inherent statistical properties of datasets and preserve data utility. This challenge becomes especially more noticeable when considering that, in this context, **the significance of individual data tuples is secondary to the underlying distribution they collectively represent** [19]. Achieving the goal of preserving these statistical properties requires a method to quantify the distance between the distributions of the original and repaired data. Traditional criteria in databases, such as subset minimality and minimum cardinality repair, often fall short in effectively addressing this requirement [8]. While various methods exist for measuring the distance between probability distributions, including information theoretic measures like Kullback-Leibler (KL) and Jensen-Shannon (JS) divergences [17], **Optimal Transport (OT) metrics, such as the Wasserstein (or Earth Mover's) distance, have demonstrated their superiority in various ML tasks** [7, 23].

OT provides a metric for comparing probability distributions by determining the most efficient way to convert one distribution into another. This transformation is facilitated through the use of a **transport plan**, which is a probabilistic mapping that specifies how much mass is moved from each data point in one distribution to its corresponding point in the second distribution. This mapping is optimized according to a designated cost function. One distinctive feature of OT is its capability to transform a domain-specific metric between individual data points into a comprehensive metric between entire distributions [7]. This adaptability empowers OT to preserve the topological and structural properties of the data that cannot be captured and maintained using other divergences and distances between distributions.

In our paper, we introduce OTClean, a novel framework that leverages OT theory for data cleaning to enforce CI constraints. OTClean addresses datasets that violate CI constraints by learning a *probabilistic data cleaner*. This cleaner probabilistically updates attribute values to ensure adherence to CI constraints. It finds an optimal repair, aiming to satisfy the CI constraint while minimizing the OT distance from the original dataset, which indicates minimal alteration to the data. This approach is versatile, allowing for user-defined metrics to tailor cleaning to specific needs and preserving data integrity, which is crucial for subsequent applications. Additionally, OTClean's probabilistic mapping operates at the tuple level, making it well-suited for streaming environments and scenarios that require model retraining on newly acquired data.

A primary hurdle in employing OT in ML is its considerable computational cost. Specifically, for discrete data, OT necessitates solving a linear program. Techniques like the network simplex or interior point methods are frequently applied, but their computational intensity is significant for high-dimensional data. In fact, their cost scales as $O(d^3 \log(d))$ when comparing histograms of dimension d [42]. **We demonstrate that using OT, the problem of repairing data under CI constraints can be formulated as a Quadratically Constrained Linear Program (QCLP)** [11, 53]. Although this problem can be tackled using established optimization techniques, it is important to note that solving a QCLP is generally NP-hard, presenting challenges in terms of scalability and computational feasibility for high-dimensional datasets.

To address the scalability challenges, we propose the use of approximate algorithms for solving our repair problem efficiently. At the core of our approach is the Sinkhorn distance [18], an approximate OT metric that introduces entropy regularization, penalizing transport plans based on their entropy. This regularization intuitively smoothens the OT problem, making it more

manageable. Importantly, it allows us to leverage Sinkhorn's matrix scaling algorithm [50], which operates at speeds several orders of magnitude faster than conventional methods. Expanding on this, **we formulate our repair problem as a regularized optimization problem that employs a relaxed version of OT along with entropic regularization**. This optimization problem remains non-convex; however, **we have developed an alternating algorithm with guaranteed convergence.** Remarkably, our approach exhibits a substantial improvement in efficiency compared to the QCLP formulation, making it scalable to high-dimensional data.

To assess the effectiveness of our approach, we apply it to two distinct domains: algorithmic fairness [47], where CI constraints play a crucial role, and data cleaning, where the utilization of CI as a statistical constraint has proven to be beneficial [56]. Our experiments reveal that our techniques outperform the current state-of-the-art database repair methods that involve CI [47]. In the realm of algorithmic fairness, **our approach not only yields fairer algorithms but also maintains superior performance compared to baseline methods**. As for data cleaning, our findings demonstrate that **enforcing CI constraints results in more accurate data representations, thereby helping prevent ML models from relying on spurious correlations.** Furthermore, we have shown that our methods can complement existing data cleaning techniques and address their limitations by effectively removing spurious correlations.

2 BACKGROUND

The notation used is summarized in Table 1. We use uppercase letters (X, Y, Z, V) to denote variables and lowercase letters (x, y, z, v) to represent their values. When referring to sets of variables or values, we use boldface notation ($\mathbf{X}$ or $\mathbf{x}$). The *support* or *domain* of a variable $\mathbf{V}$ is given by $\mathcal{V}$. We use $d_{\mathcal{V}}$ to refer to $|\mathcal{V}|$, i.e., the size of $\mathcal{V}$'s support. For any discrete random variable X, its probability distribution is represented by $P_X(x)$; in some contexts, we might simply use P, indicating the probability of X assuming the value x. It's essential to note that such a probability distribution P can be equivalently seen as a point in the *probability Simplex* $\Delta_{\mathbf{V}} = \{\mathbf{X} \in \mathbb{R}^{d_{\mathcal{V}}} \mid \forall v \in \mathcal{V}, \mathbf{X}_v \geq 0 \text{ and } \sum_{v \in \mathcal{V}} \mathbf{X}_v = 1\}$, where, $\mathbf{X}_v$ is the probability assigned to value v. Intuitively, $\Delta_{\mathbf{V}}$ defines the set of all possible probability distributions over the finite domain $\mathcal{V}$.

Given a probability distribution $P \in \Delta_{\mathbf{V}}$ over a set of variables $\mathbf{V}$, and considering non-empty and disjoint subsets $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ within $\mathbf{V}$, the distribution P is said to be *consistent* with a *conditional independence (CI) constraint* ($\sigma : \mathbf{Y} \perp\!\!\!\perp \mathbf{X} \mid \mathbf{Z}$), denoted as $P \models \sigma$, if and only if, for all values $x \in \mathcal{X}$, $y \in \mathcal{Y}$, and $z \in \mathcal{Z}$, the condition $P_{\mathbf{X},\mathbf{Y}|\mathbf{Z}}(x, y \mid z) = P_{\mathbf{X}|\mathbf{Z}}(x \mid z) \cdot P_{\mathbf{Y}|\mathbf{Z}}(y \mid z)$ is satisfied. If the entire set $\mathbf{V}$ is precisely the union of the subsets $\mathbf{X}, \mathbf{Y}$, and $\mathbf{Z}$, i.e., $\mathbf{V} = \mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}$, then the constraint σ is termed as *saturated*.

When P is *inconsistent* with the constraint $\sigma : Y \perp\!\!\!\perp X \mid Z$, the *degree of inconsistency* of P, denoted $\delta_\sigma(P)$, can be quantified using the *conditional mutual information (CMI)*, denoted as $I(X; Y \mid Z)$, which measures the amount of information about Y obtained by knowing X, given Z. Formally,

$$
\begin{aligned}
I(X; Y \mid Z) &= \sum_{x \in \mathcal{X}, y \in \mathcal{Y}, z \in \mathcal{Z}} P(x, y, z) \log \left(\frac{P_{X,Y|Z}(x, y \mid z)}{P_{X|Z}(x \mid z) P_{Y|Z}(y \mid z)} \right) \\
&= D_{\mathrm{KL}}[P(X, Y, Z) \mid P(X, Z) P(Y \mid Z)]
\end{aligned}
$$

where D_{KL} is the Kullback–Leibler divergence[1].
The probability distribution P is consistent with the constraint $\sigma : Y \perp\!\!\!\perp X \mid Z$ if and only if $I(X; Y \mid Z) = 0$.

[1]The Kullback–Leibler divergence between two distribution $Q(X)$ and $P(X)$ is defined as: $D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right)$.

Table 1. Summary of notation and symbols.

Symbol	Description
X, Y, Z, V	Variables
$\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{V}$	Sets of variables
$\mathcal{X}$	Domain of a variable X
$d_{\mathcal{X}}$	Size of the domain of a variable X
$x \in \mathcal{X}$	Their values
P	Probability distributions
$\Delta_{\mathbf{V}}$	A probability simplex over a domain of variables $\mathbf{V}$
$\mathbf{p} \in \Delta_{\mathbf{V}}$	A probability vector
π	Transport plan
$\sigma : (X \perp\!\!\!\perp Y \mid Z)$	A CI constraint
$\delta_\sigma(P)$	Degree of inconsistency of P to a CI constraint σ
$c, \mathbf{C}$	Cost function and cost matrix

Given a dataset $D = \{\mathbf{v}_i\}_{i=1}^n$ consisting of i.i.d. samples drawn from a distribution $P \in \Delta_{\mathbf{V}}$, each sample $\mathbf{v}_i$ corresponds to an element in the domain $\mathcal{V}$. The *empirical distribution* P^D of the dataset D is defined as: $P^D_{\mathbf{V}}(\mathbf{v}) = \frac{1}{n}\sum_{i=1}^n \mathbb{I}(\mathbf{v}_i = \mathbf{v})$, where $\mathbb{I}$ is the indicator function that returns 1 if its argument is true and 0 otherwise. For each value $\mathbf{v}$ in the domain $\mathcal{V}$, $P^D_{\mathbf{V}}(\mathbf{v})$ computes the fraction of times $\mathbf{v}$ appears in the dataset D. This empirical distribution provides an estimate of the true underlying distribution P from which the samples in D were drawn. Given a conditional independence constraint $\sigma : \mathbf{Y} \perp\!\!\!\perp \mathbf{X} \mid \mathbf{Z}$, we say D is consistent with σ if the empirical distribution P^D associated with D is consistent with it. This is also denoted as $D \models \sigma$.

2.1 Background on Optimal Transport

This section provides an overview of optimal transport, serving as the foundational theory for OTClean. We further delve into Sinkhorn regularization and the concept of relaxed optimal transport, which underpin the approximate repair methods introduced in Section 4.2.

Monge problem: *The Optimal Transport (OT) problem* seeks the most efficient way of transferring mass from a probability distribution P to another while preserving the total mass. The OT problem's classical formulation is *the Monge problem* where the objective is to identify a *transport map* T that pushes a distribution $P \in \Delta_{\mathbf{X}}$ forward to a distribution $Q \in \Delta_{\mathbf{Y}}$ while minimizing the total cost of transporting mass. Formally, Q, known as *the pushforward* of P under the transport map T, is a new distribution defined as $Q(A) = P(T^{-1}(A))$ for any $A \subseteq \mathcal{Y}$. In other words, the pushforward Q characterizes the distribution of the images of P under the map T. The Monge problem can be formally defined as follows: Given two distributions P and Q with discrete supports $\mathcal{X}$ and $\mathcal{Y}$, respectively, and a cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$, the goal is to find a transport map $T : \mathcal{X} \to \mathcal{Y}$ that pushes forward P to Q, such that the total cost of transporting mass is minimized, i.e.,

$$\mathrm{OT}_{Monge}(P, Q) = \underset{T:\mathcal{X}\to\mathcal{Y}}{\operatorname{argmin}} \sum_{\mathbf{x}_i \in \mathcal{X}} c(\mathbf{x}_i, T(\mathbf{x}_i)), \tag{1}$$

where T is a transport map and $T_\# P = Q$.

Kantorovich Formulation. The deterministic transport approach in Monge's problem might not always admit a solution. Specifically, there may be cases where finding a pushforward between two distinct probability distributions is not feasible. To overcome this limitation, Kantorovich introduced a more flexible formulation by considering probabilistic transport methods. Unlike the deterministic approach, which requires a direct one-to-one mapping between elements, probabilistic

transport allows for a more versatile mapping where elements from one distribution can be mapped to multiple elements in another distribution, reflecting real-world scenarios where such distributions cannot always be perfectly aligned. This approach is operationalized through the concept of *transport plans* or *couplings*. Here, a coupling refers to a joint distribution, denoted as π, over the product space $\mathcal{X} \times \mathcal{Y}$. This coupling ensures that its marginals match the given distributions P and Q, meaning $P = \pi(\mathbf{X})$ and $Q = \pi(\mathbf{Y})$. Denote $\Pi(P, Q)$ as the space of all possible couplings. In this context, the *primal Kantorovich formulation* of the OT problem is defined as follows:

$$\mathrm{OT}(P, Q) = \underset{\pi \in \Pi(P,Q)}{\operatorname{argmin}} \sum_{\mathbf{x}_i \in \mathcal{X}} \sum_{\mathbf{y}_j \in \mathcal{Y}} c(\mathbf{x}_i, \mathbf{y}_j)\pi(\mathbf{x}_i, \mathbf{y}_j). \tag{2}$$

The goal of the OT plan π is to minimize the overall transport cost, as expressed in Equation 2, while adhering to the probabilistic nature of the transport. When the cost c represents the Euclidean distance, the OT distance is recognized as the *Wasserstein distance*.

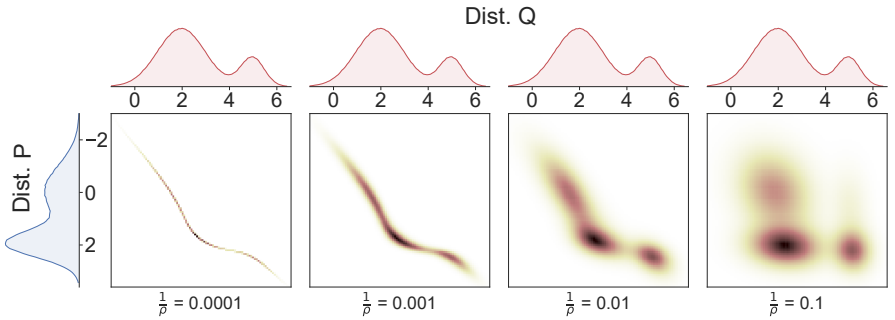


Fig. 1. The coefficient $1/\rho$ in regularized OT impacts the mapping between distributions P and Q: higher coefficients (on the right) lead to smoother mappings and spread mass more evenly between P and Q.

Entropic Regularization: OT problems, as described by Equation 2, essentially involve solving a linear program. The computational complexity of solving such a linear program $O(n^3 \log n)$ using the network simplex, where n represents the number of variables or constraints [42]. This complexity can become a significant challenge, especially for high-dimensional datasets. To mitigate this computational burden, entropic regularization has been introduced as an effective strategy [18]. By incorporating an entropy term into the optimal transport formulation, the problem is transformed into a nonlinear but *smooth* optimization problem, which can be solved more efficiently. This adjustment not only reduces the complexity of the problem but also enables its solution using linear-time algorithms. In the case of entropic regularization, the added entropy term effectively spreads out the transport plan, preventing the concentration of mass in a few narrow pathways. This spreading leads to a more evenly distributed plan, reducing the presence of sharp peaks and troughs in the optimization landscape. As a result, the optimization problem becomes more regular, with a smoother surface that is easier to navigate using optimization algorithms.

In more formal terms, the entropic OT is defined by:

$$\underset{\pi \in \Pi(P,Q)}{\operatorname{argmin}} \sum_{\mathbf{x}_i \in \mathcal{X}} \sum_{\mathbf{y}_j \in \mathcal{Y}} c(\mathbf{x}_i, \mathbf{y}_j)\pi(\mathbf{x}_i, \mathbf{y}_j) - \frac{1}{\rho} H(\pi). \tag{3}$$

where $H(\pi)$ is the entropic regularizer:

$$H(\pi) = -\sum_{\mathbf{x}_i \in \mathcal{X}} \sum_{\mathbf{y}_j \in \mathcal{Y}} \pi(\mathbf{x}_i, \mathbf{y}_j) \log(\pi(\mathbf{x}_i, \mathbf{y}_j))$$

and $1/\rho$ is the *entropic regularization parameter*. A smaller value means that we emphasize the accuracy of the transport plan, while a larger value leans towards computational efficiency.

Importantly, the OT plan π^*, which solves the constrained optimization problem defined in (3), manifests as a diagonal scaling of the matrix $\mathbf{K} \coloneqq e^{-\frac{\mathbf{C}}{\rho}}$. Specifically, it has been shown that the solution to (3) is unique and takes the form $\pi^* = \text{diag}(\mathbf{u}) \cdot \mathbf{K} \cdot \text{diag}(\mathbf{v})$, with $\mathbf{u}$ and $\mathbf{v}$ acting as scaling vectors. These scaling vectors are identified through an iterative process, which ensures that the resultant transport plan complies with marginal probability constraints. The Sinkhorn Algorithm, crucial for this process, iteratively adjusts $\mathbf{u}$ and $\mathbf{v}$ to ensure that the resultant transport matrix, π^*, adheres to the given marginal constraints. Lines 4 and 5 of Algorithm 1 represent these adjustments. Specifically, $\mathbf{u}$ and $\mathbf{v}$ are updated iteratively to balance the rows and columns of $\mathbf{K}$, ensuring that the marginals of the scaled coupling matrix π closely match $\mathbf{p}$ and $\mathbf{q}$.

Algorithm 1: Sinkhorn Algorithm

Input: Probability distributions P, Q and cost function c
Output: A transport plan between P and Q

1 $\mathbf{p} \coloneqq vector(P); \mathbf{q} \coloneqq vector(Q); \mathbf{C} \coloneqq matrix(c);$
2 $\mathbf{u} \coloneqq \mathbb{1}_{d_X}; \mathbf{v} \coloneqq \mathbb{1}_{d_Y}; \mathbf{K} \coloneqq e^{-\frac{\mathbf{C}}{\rho}};$ ▷ Initialization
3 **while** $\boldsymbol{u}$ *and* $\boldsymbol{v}$ *are not converged* **do** ▷ Sinkhorn iterations
4 $\quad \mathbf{u} \coloneqq \mathbf{p} \oslash (\mathbf{K} \cdot \mathbf{v});$ ▷ $\oslash$: Element-wise division
5 $\quad \mathbf{v} \coloneqq \mathbf{q} \oslash (\mathbf{K} \cdot \mathbf{u});$
6 $\pi \coloneqq diag(\mathbf{u}) \cdot \mathbf{K} \cdot diag(\mathbf{v});$
7 **return** π;

Example 2.1. Figure 1 presents the optimal transport between two Gaussian mixture model distributions, P and Q. Each distribution is a mixture of two Gaussians, providing a basis for examining the effects of entropic regularization on transport plans. The leftmost graph in Figure 1 shows the original OT plan without entropic regularization. The optimal plan is more deterministic and sharp in mapping elements between the distributions. As we introduce and increase the entropic regularization coefficient, the subsequent transport plans become more spread out. This spread is visually observable in Figure 1, where higher coefficients lead to transport plans that are less focused and more distributed across the space. This effect illustrates the principle of entropic regularization: a lower coefficient results in a transport plan that closely aligns specific elements of the distributions, whereas a higher coefficient allows for a broader, more generalized mapping. The intuition behind these transport plans can be understood by considering how the elements of one distribution, say ranging between -2 and 3 in P, might be transported to another distribution Q with values ranging between 0 and 6. Without regularization, the transport plan seeks to map these elements in a direct and specific manner. However, with entropic regularization, the mapping allows for the mass from one value in P to be spread across the target distribution and to be transported to many values in P, thereby avoiding overly precise mappings that might not generalize well across different scenarios. This approach is particularly useful when dealing with high-dimensional data, where overly specific mappings can lead to overfitting and reduced model robustness.

Relaxed Optimal Transport: Relaxed OT, introduced in [23], provides a loss function for supervised learning grounded in OT principles. Rather than relying on hard marginal constraints typical of entropic regularized OT, it adopts softer penalties, using regularization based on the Kullback-Leibler (KL) divergence. This approach leads to:

$$\underset{\pi \in \mathcal{J}}{\operatorname{argmin}} \sum_{\mathbf{x}_i \in \mathcal{X}} \sum_{\mathbf{y}_j \in \mathcal{Y}} c(\mathbf{x}_i, \mathbf{y}_j)\pi(\mathbf{x}_i, \mathbf{y}_j) - \frac{1}{\rho} H(\pi) + \lambda(D_{\mathrm{KL}}(\pi(Y), Q) + D_{\mathrm{KL}}(\pi(X), P)). \tag{4}$$

where λ is the relaxation regularization coefficient, and D_{KL} denotes the KL divergence between two probability distributions. Contrasting this with the entropic OT outlined in Equation 3, the transport plan π in relaxed OT can be an element of $\mathcal{J}$, which includes all possible joint probability distributions over the product space $\Delta_{\mathbf{X}} \times \Delta_{\mathbf{Y}}$. It has been shown in [23] that Sinkhorn algorithm also works for the relaxed version of the entropic OT in Equation 3 but with different update rules for $\mathbf{u}$ and $\mathbf{v}$ [23, Proposition 4.2]:

$$\mathbf{u} = (\mathbf{p} \oslash (\mathbf{K} \cdot \mathbf{v}))^{\frac{\rho\lambda}{\rho\lambda+1}} \quad \text{and} \quad \mathbf{v} = (\mathbf{q} \oslash (\mathbf{K}^\top \cdot \mathbf{u}))^{\frac{\rho\lambda}{\rho\lambda+1}} \tag{5}$$

3 PROBLEM DEFINITION

Given a database D that is inconsistent with a CI constraint $\sigma : (\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z})$, our objective is to resolve this inconsistency by updating the attribute values of each datapoint in D to derive a repaired database $\hat{D}$ which is consistent with σ. To ensure minimal distortion and maintain the utility of the data, we assume we are given a user-defined cost function that quantifies the cost of updating a datapoint (this cost function generalizes the minimality criteria in update-based data repair in databases [8]). Leveraging the principles of OT, our goal is to develop a data cleaner, envisioned as a transport map, that repairs D at a minimum cost. Next, we define the problem of learning an optimal data cleaner for a CI constraint.

Definition 3.1 (CI Data Cleaner). Consider a database $D = \{\mathbf{v}_i\}_{i=1}^n$ that violates a CI constraint σ, i.e., $D \not\models \sigma$, and a user-defined cost function $c : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0}$ that assigns a cost to transforming or perturbing one tuple in $\mathcal{V}$ to another tuple in $\mathcal{V}$. The CI data cleaner of D with respect to σ is a transport map $T^* : \mathcal{V} \to \mathcal{V}$ that transforms D into a database $\hat{D} = T^*(D) = \{\hat{\mathbf{v}}_i = T^*(\mathbf{v}_i)\}_{i=1}^n$ such that $\hat{D} \models \sigma$ and has the minimum transportation cost, i.e., T^* is the solution to the following constrained optimization problem:

$$\arg\min_{T} \sum_{i=1}^{n} c(\mathbf{v}_i, T(\mathbf{v}_i)) \qquad \textit{s.t.} \quad T(D) \models \sigma. \tag{6}$$

We illustrate an optimal data cleaner with an example:

Example 3.2. Let's consider a database $D_1 = \{(0,0,1), (1,0,1), (0,1,1), (0,1,0)\}$ defined over binary variables X, Y, and Z. D_1 violates the CI constraint $\sigma : Y \perp\!\!\!\perp Z$ because the probability $P_{Y,Z}(1,0)$ is $\frac{1}{4}$, which is not equivalent to the product of the marginal probabilities $P_Y(1) = \frac{2}{4}$ and $P_Z(0) = \frac{1}{4}$. Further, suppose cost is measured using Euclidean distance. An optimal CI repair can be obtained using the transport map T, which maps $(0,0,1) \to (0,0,0)$ and other tuples to their current values. As a result, by updating one attribute value, T transforms D_1 into a repaired database $\hat{D}_1 = \{(0,0,0), (1,0,1), (1,1,0), (0,1,1)\}$, which is consistent with σ.

However, the CI data cleaner defined in Definition (3.1) might not lead to a minimum cost repair. This is especially true if D is a bag, which is typically the case with databases used for ML. These databases are either bags or projections onto a subset of features that yield a bag. We illustrate this with an example:

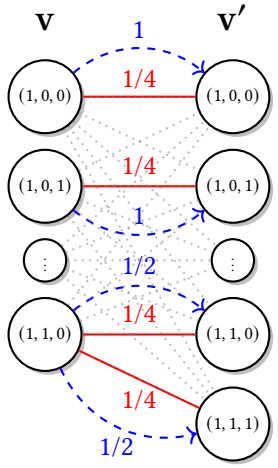
Example 3.3. Continuing with Example 3.2, now consider a database $D_2 = \{(1,0,0), (1,0,1), (1,1,0), (1,1,0)\}$*, which is now a bag, and is inconsistent with the constraint* $Y \perp\!\!\!\perp Z$*. Similarly,* $\hat{D}_2 = \{(1,0,0), (1,0,1), (1,1,0), (1,1,1)\}$ *is a minimum cost repair for* D_2*, obtained by modifying only one attribute value. However, no transport map exists that can transport* D_2 *into* $\hat{D}_2$ *simply because* $(1,1,0)$ *cannot be mapped to both itself and* $(1,1,1)$*. Upon close examination, it becomes evident that no transport map can lead to a repair for* D_2 *with cost 1.*

Probabilistic Optimal Data Cleaner. As demonstrated in Example 3.3, the transport map defined in Definition 3.1 does not always yield the minimum cost repair (although it can always produce a trivial repair by mapping every tuple to a single tuple, which completely distorts the distribution). Indeed, it's possible for the minimum cost repair to be outside the feasible region defined by the problem in Equation (6). Drawing from the Kantorovich relaxation of OT, we shift our approach to seeking a transport plan, or transport coupling, denoted as $\pi(\mathbf{v}', \mathbf{v})$, as an alternative to a deterministic transport map T. Here, the marginal distribution $\pi(\mathbf{v}) = P^D$ represents the empirical distribution of the database D, and $\pi(\mathbf{v}')$ is the target distribution that is consistent with the CI constraint. This transport plan yields a probabilistic mapping, $\pi(\mathbf{v}' \mid \mathbf{v})$, which probabilistically updates a data point $\mathbf{v} \in D$ to $\mathbf{v}'$ following the mapping. The repaired database is then obtained by applying this mapping to D, by sampling. In essence, Definition 3.1 transitions into a problem where the aim is to (1) identify a transport plan $\pi(\mathbf{v}', \mathbf{v})$ that pushforwards the distribution $\pi(\mathbf{v}) = P^D$, i.e., the empirical distribution associated with D into one consistent with the CI constraints, and (2) among all distributions with the same support and consistent with the constraint, find the distribution $\pi(\mathbf{v}')$ with the minimum OT distance to $\pi(\mathbf{v}) = P^D$. Formally, an *optimal probabilistic data cleaner* for CI constraint seeks to clean data using a probabilistic mapping $\pi(\mathbf{v}' \mid \mathbf{v})$ associated with a transport plan or probabilistic coupling $\pi(\mathbf{v}', \mathbf{v})$, obtained by solving the following optimization problem:

$$\arg\min_{\pi} \sum_{i=1}^{d_{\mathcal{V}}} \sum_{j=1}^{d_{\mathcal{V}}} c(\mathbf{v}_i, \mathbf{v}'_j)\pi(\mathbf{v}_i, \mathbf{v}'_j) \;\text{ s.t. }\; \pi(\mathbf{v}) = P^{\mathrm{D}},\; \pi(\mathbf{v}') \models \sigma. \tag{7}$$

The feasible region of the optimization problem defined in Equation 7 consists of all possible probability distributions that satisfy the constraint, hence including a distribution associated with a minimal cost repair. Therefore, one can find a mapping that transforms the empirical distribution of D into a consistent distribution with the minimum cost. Moreover, the optimal probabilistic mapping, derived from solving Equation 7, provides an approach for probabilistic data cleaning. For large datasets, samples drawn from this probabilistic cleaner will lead to a dataset $\hat{D}$ whose empirical distribution $P^{\hat{D}}$ closely aligns with the target distribution $P(\mathbf{v}')$, in line with the law of large numbers. Consequently, the resulting dataset is approximately consistent with the constraint. In ML applications, this level of approximation is generally adequate.

Example 3.4. Consider $D_2 = \{(1,0,0), (1,0,1), (1,1,0), (1,1,0)\}$ *from Example 3.3. The probabilistic mapping* $\pi(\mathbf{v}, \mathbf{v}')$ *is graphically represented in Figure 2, which depicts the bipartite graph constructed from the elements of the domain* $\mathcal{V}$*. Labeled red edges illustrate the joint probabilities* $\pi(\mathbf{v}, \mathbf{v}')$*, while dashed directed edges showcase the corresponding probabilistic mapping* $\pi(\mathbf{v} \mid \mathbf{v}')$*. The graph only includes nodes and edges for which* $\pi(\mathbf{v}, \mathbf{v}')$ *and* $\pi(\mathbf{v} \mid \mathbf{v}')$ *are non-zero to maintain clarity. It's evident that the marginal distribution* $\pi(\mathbf{v})$ *displayed in Figure 2 matches the empirical distribution* P^{D_2} *associated to* D_2*. Furthermore,* $\pi(\mathbf{v} \mid \mathbf{v}')$ *primarily maps all elements to themselves with a probability of 1. However, it transports half of the mass from* $(1,1,0)$ *to itself and the other half to* $(1,1,1)$ *to repair the constraint violation. This results in a distribution* $\pi(\mathbf{v}')$ *consistent with the constraint. Notably, the OT cost of this repair is* $1/4$ *since just* $1/4$ *of the mass with cost 1 transitions from* $(1,1,0)$ *to* $(1,1,1)$*.*



v	$\pi(\mathbf{v})$
(1, 0, 0)	1/4
(1, 0, 1)	1/4
(1, 1, 0)	1/2

v′	$\pi(\mathbf{v}')$
(1, 0, 0)	1/4
(1, 0, 1)	1/4
(1, 1, 0)	1/4
(1, 1, 1)	1/4

Fig. 2. Graphical representation of the plan $\pi(\mathbf{v}, \mathbf{v}')$ for D_2. Nodes represent elements in $\mathcal{V}$. Labeled red edges indicate joint probabilities $\pi(\mathbf{v}, \mathbf{v}')$, while dashed directed edges depict the probabilistic mapping $\pi(\mathbf{v} \mid \mathbf{v}')$. Only nodes and edges with non-zero probabilities are shown for clarity.

The mapping $\pi(\mathbf{v} \mid \mathbf{v}')$ can be employed to clean D_2 probabilistically. Due to the limited sample size, this doesn't guarantee consistency. Still, for a larger database, the repaired database becomes representative of $\pi(\mathbf{v}')$ and hence becomes consistent with the constraint. To illustrate this, consider another database D_3 echoing the tuples in D_2, but each tuple is now replicated n times. This mirrors the empirical distribution of D_2 and still violates the constraint. In such a scenario, repairing D_3 with $\pi(\mathbf{v} \mid \mathbf{v}')$ likely results in a consistent database. Probabilistically repairing the $2n$ instances of $(1, 1, 0)$ in D_3 through the mapping $\pi(\mathbf{v}' \mid \mathbf{v})$ can be interpreted as a sequence of $2n$ Bernoulli trials with a 1/2 probability. On average, this yields n tuples of $(1, 1, 0)$ and n tuples of $(1, 1, 1)$, ensuring consistency with the constraints.

Discussion on Complexity. Designing scalable algorithms to solve the optimization problem outlined in (7) and subsequently computing optimal repairs for CI constraints presents significant challenges. A straightforward approach entails exploring the vast space of all distributions consistent with the CI, computing OT distance in relation to the empirical distribution of D, and identifying the optimal solution. This method, however, is not feasible primarily due to the intractable nature of the space of consistent distributions. Furthermore, as discussed in 1, the computation of OT is computationally demanding. In our context, the transport plan involves $d^2_{\mathcal{V}}$ variables, thereby exacerbating the inherent complexity.

Although a detailed complexity analysis of the optimization problem 7 is not addressed in this paper, it is worth noting that our problem is akin to the computation of minimum update-based repair (U-repair) for MVDs [8]. U-repair aims to identify a repair that necessitates the fewest attribute value modifications to enforce an MVD. Specifically, given a database D with attributes XYZ and an MVD $X \twoheadrightarrow Y$, the decision problem is whether D has an optimal U-repair with no more than k modifications. This decision problem can be translated to our repair challenge by presuming a uniform distribution over D, considering a cost function $c(x, y, z, x', y', z')$ that enumerates the number of modifications required to obtain (x', y', z') from (x, y, z), and checking if D can achieve an optimal repair at a cost lesser than k given the conditional independence $X \perp\!\!\!\perp Y \mid Z$. Under the specified assumptions, it is easy to check $D \models (X \perp\!\!\!\perp Y \mid Z)$ if and only if $D \models X \twoheadrightarrow Y$. While there's

$$\pi(\mathbf{v}, \mathbf{v}')$$

$$P^{D_2}(\mathbf{v}) = \pi(\mathbf{v}) \begin{bmatrix} \tilde{\pi}_{1,1} & \cdots & \tilde{\pi}_{1,7} & \tilde{\pi}_{1,8} \\ \tilde{\pi}_{2,1} & \cdots & \tilde{\pi}_{2,7} & \tilde{\pi}_{2,8} \\ \tilde{\pi}_{3,1} & \cdots & \tilde{\pi}_{3,7} & \tilde{\pi}_{3,8} \end{bmatrix} \tilde{Q}(\mathbf{v}') = \pi(\mathbf{v}')$$

$\mathbf{v}'$	$\tilde{Q} = \pi(\mathbf{v}')$
$(0,0,0)$	$\tilde{Q}(0,0,0)$
$\vdots$	$\vdots$
$(1,1,0)$	$\tilde{Q}(1,1,0)$
$(1,1,1)$	$\tilde{Q}(1,1,1)$

Objective:

$$\min_{\tilde{\pi}}(1\times\tilde{\pi}_{1,1} + 2\times\tilde{\pi}_{1,2} + 2\times\tilde{\pi}_{2,1} + \ldots + 1\times\tilde{\pi}_{3,8})$$

Marginal constraints:

$$\tilde{\pi}_{1,1} + \tilde{\pi}_{1,2} + \cdots + \tilde{\pi}_{1,8} = \tfrac{1}{4}$$
$$\tilde{\pi}_{2,1} + \tilde{\pi}_{2,2} + \cdots + \tilde{\pi}_{1,8} = \tfrac{1}{4}$$
$$\tilde{\pi}_{3,1} + \tilde{\pi}_{3,2} + \cdots + \tilde{\pi}_{3,8} = \tfrac{1}{2}$$

Validity constraints:

$$\tilde{\pi}_{1,1} \geq 0, \tilde{\pi}_{1,2} \geq 0, \cdots, \tilde{\pi}_{3,8} \geq 0$$

Independence constraints:

$$\tilde{Q}_{Y,Z}(0,0) = \tilde{Q}_Y(0) \times \tilde{Q}_Z(0)$$
$$\tilde{Q}_{Y,Z}(0,1) = \tilde{Q}_Y(0) \times \tilde{Q}_Z(1)$$
$$\tilde{Q}_{Y,Z}(1,0) = \tilde{Q}_Y(1) \times \tilde{Q}_Z(0)$$
$$\tilde{Q}_{Y,Z}(1,1) = \tilde{Q}_Y(1) \times \tilde{Q}_Z(1)$$

Fig. 3. The QCLP for Example 4.1. The top left is the transport plan defined by the decision variables. The top right is $\tilde{Q}$ definitions. The rest are the objective and constraints.

extensive literature on the U-repair problem for Functional Dependencies [33, 38], to the best of our knowledge, it hasn't been studied for MVDs.

4 EFFICIENT COMPUTATION OF PROBABILISTIC OPTIMAL DATA CLEANER

In this section, we introduce efficient methods for computing the optimal data cleaner for CI constraints as described in (7). In Section 4.1, we formulate the problem as a Quadratically Constrained Linear Program (QCLP). This formulation allows for the derivation of an exact solution using existing efficient algorithms designed for QCLP. Subsequently, in Section 4.2, we present an approximate version of the optimization problem in (7). This approach facilitates the development of scalable and efficient solutions using iterative algorithms, particularly those based on Sinkhorn's matrix scaling.

4.1 QCLP Formulation

We present a QCLP designed to find an optimal data cleaner, as outlined in Section 3. This program takes three inputs: a database D, a CI constraint σ, and a cost function c. We assume that σ is a saturated CI constraint (i.e., it contains all attributes of D cf. 2), with discussions on extending to unsaturated CI in Section 5.

To formulate the QCLP, we first describe the decision variables in the program, followed by an explanation of the constraints and the objective function. For clarity and better understanding, we use D_2 from Example 3.4 to demonstrate the QCLP formulation.

Decision Variables. In the QCLP, decision variables are represented as $\tilde{\pi}_{i,j}$, where both i and j span from 1 up to $d_{\mathcal{V}}$ (reflecting the size of the support of $\mathcal{V}$). These variables are the transport plan's probabilities representing the optimal data cleaning strategy. Since this plan has non-zero probabilities exclusively for the values present in D's active domain, i's range can be limited to the size of D's active domain. The following example clarifies this.

Example 4.1. In the QCLP for the optimal cleaner of D_2 from Example 3.4, the transport plan is defined by an 8×8 variable matrix. However, given that D_2 contains only three records, we use a 3×8 decision variable matrix, with the remaining rows of the initial matrix being zero. These decision variables indicate possible modifications to the three records in D_2, enabling them to align with any of

the eight potential records in $\hat{D}_2$. The QCLP considers all eight potential records in $\hat{D}_2$, each associated with its distinct variable.

Constraints. The QCLP incorporates three types of constraints to encode the conditions in our data cleaner formulation in (7):

- *Validity Constraints:* These constraints, together with marginal constraints, ensure that $\tilde{\pi}$ makes a valid transport plan. Specifically, the decision variables must be non-negative real values:

$$\tilde{\pi}_{i,j} \geq 0 \quad \forall i \in [1, d_{\mathcal{V}}], j \in [1, d_{\mathcal{V}}] \tag{8}$$

- *Marginal Constraints:* These constraints are included to guarantee that the marginals of the transport plan, as described by $\tilde{\pi}$, align with P^D (the empirical distribution of D):

$$\sum_{j=1}^{d_{\mathcal{V}}} \tilde{\pi}_{i,j} = P^D(\mathbf{v}_i) \quad \forall i \in [1, d_{\mathcal{V}}] \tag{9}$$

- *Independence Constraints:* These constraints are formulated to ensure that the probability distribution $\pi(\mathbf{v}')$ satisfies the CI constraint $\sigma : (X \perp\!\!\!\perp Y \mid Z)$. To express these constraints, we introduce $\tilde{Q}$ as the marginal probability distribution obtained from the decision variables $\tilde{\pi}$. The independence constraints express the equation $\tilde{Q}_{X,Z}(x', z') \times \tilde{Q}_{Y,Z}(y', z') = \tilde{Q}(x', y', z') \times \tilde{Q}_Z(z')$ and guarantee the marginal probability distribution satisfies σ. We use the notation $\tilde{Q}$ instead of Q to emphasize that the decision variables in $\tilde{\pi}$ specify the marginal probability distribution.

Objective. The objective of the QCLP is to minimize the transport cost, which is represented as follows:

$$\min_{\tilde{\pi}} \sum_{i=1}^{d_{\mathcal{V}}} \sum_{j=1}^{d_{\mathcal{V}}} c(\mathbf{v}_i, \mathbf{v}j) \times \tilde{\pi}_{i,j} \tag{10}$$

In this expression, the transport cost is calculated by summing the product of the cost function $c(\mathbf{v}_i, \mathbf{v}j)$ and the decision variables $\tilde{\pi}_{i,j}$, over all elements in the set $\mathcal{V}$.

Example 4.2. Expanding on Example 4.1, Figure 3 shows the constraints and objective present in the QCLP for D_2. Specifically, the validity constraints ensure that 24 decision variables are non-negative. The three marginal constraints verify the alignment of the marginal probability, as defined by the transport plan, with the probabilities of the three input records in D_2. The independence constraints ensure that the probability distribution specified by $\tilde{Q}$ satisfies $\sigma : X \perp\!\!\!\perp Y \mid Z$. For example, four independence constraints in this example guarantee $\sigma : Y \perp\!\!\!\perp Z$ holds for all possible values of Y and Z. The first independence constraint is $\tilde{Q}_{Y,Z}(0,0) = \tilde{Q}_Y(0) \times \tilde{Q}_Z(0)$, where the marginals $\tilde{Q}_{Y,Z}(0,0), \tilde{Q}_Y(0)$, and $\tilde{Q}_Z(0)$ are defined as sums of decision variables in $\tilde{\pi}$. The costs in the objective are the Euclidean distance between the input records and their possible repair, e.g., the cost 1 in $1 \times \tilde{\pi}_{1,1}$ is the Euclidean distance between $(1,0,0)$, as the first record in D_2, and $(0,0,0)$, as the first possible repair. Similarly 2 in $2 \times \tilde{\pi}_{1,2}$ reflects the Euclidean distance between $(1,0,0)$ and $(0,0,1)$.

The above program is classified as a QCLP because, while the objective function and the validity and marginal constraints are linear with respect to the decision variables, the independence constraints are non-linear (quadratic). This is due to each side of the constraint consisting of a product of values in $\tilde{Q}$, that each is, in turn, a sum of the variables in $\tilde{\pi}$. QCLP represents a distinct subtype of Quadratically Constrained Quadratic Programs (QCQPs) or Second-Order Cone Programs (SOCPs) that feature quadratic constraints and objectives. Addressing a QCLP is a non-convex optimization problem and is NP-hard [11, 53]. Diverse, efficient methodologies, including sequential quadratic programming, augmented Lagrangian, interior-point, and active set, have been employed to derive sub-optimal solutions for such programs [11].

We implemented an alternating algorithm to compute the optimal repair by solving the QCLP program. This method iteratively transforms the quadratic independence constraints into linear ones, similar to the Alternating Direction Method of Multipliers (ADMM) [12]. The process begins with initial variable estimates for $\tilde{\pi}$, ensuring the marginal distribution $\tilde{Q}$ satisfies σ. These initial values can be derived from the marginal probabilities of P^D. In each iteration, we partition the variables in $\tilde{\pi}$ into two subsets. We substitute the variables with their current estimates for the first subset, effectively linearizing the constraints. This transformation allows us to treat the second subset as variables within a linear program. In subsequent iterations, we alternate roles: treating variables of the second subset as constants and updating the first subset's values by solving a distinct linear program. This alternating process continues until the variables stabilize, indicating convergence. We have omitted the algorithm's specifics for brevity. The algorithm's convergence proof is similar to that of ADMM as presented in [12].

4.1.1 Analysis of the QCLP Solution. The QCLP formulation, though convergent, encounters scalability challenges. Specifically, in each iteration, it necessitates solving an OT problem which is structured as a linear program. The computational complexity of determining the OT scales as $O(d^3 \log(d))$ when comparing histograms of dimension d [42]. In the following section, we introduce an alternative formulation that mitigates this scalability issue and obviates the need for solving a linear program.

4.2 Fast Approximation via Relaxed OT using Sinkhorn Iterations

In this section, we present an approximate algorithm for computing optimal repairs by casting the problem into a regularized optimization. This approach integrates the CI constraint and the constraint on marginals as regularizers, drawing inspiration from the relaxed optimal transport discussed in Section 2. Specifically, we formulate the problem of computing the optimal cleaner in (7) as the following regularized optimization problem:

$$\underset{\pi \in \Pi, Q \in \Delta_{\mathbf{V}}}{\operatorname{argmin}} \sum_{\mathbf{v}_i \in \mathcal{V}} \sum_{\mathbf{v}'_j \in \mathcal{V}} c(\mathbf{v}_i, \mathbf{v}'_j)\pi(\mathbf{v}_i, \mathbf{v}'_j) - \frac{1}{\rho} H(\pi) + \lambda(D_{\mathrm{KL}}(\pi(\mathbf{v}'), Q) + D_{\mathrm{KL}}(\pi(\mathbf{v}), P^D)) + \mu\, \delta_\sigma(Q), \tag{11}$$

In the above formulation, P^D denotes the empirical distribution of the dataset D. The target distribution, represented by Q, functions as a decision variable, while π is the transport plan. The regularization term $D_{\mathrm{KL}}(\pi(\mathbf{v}'), Q) + D_{\mathrm{KL}}(\pi(\mathbf{v}), P^D)$ penalizes the objective when there are deviations of its marginals $\pi(\mathbf{v})$ and $\pi(\mathbf{v}')$ from P^D and Q, respectively. Additionally, the CI constraint, represented by σ, is imposed on Q through the regularization term $\delta_\sigma(Q)$ within the objective (recall from Section 1 that $\delta_\sigma(Q) = D_{\mathrm{KL}}[Q(X, Y, Z) \mid Q(X, Z)Q(Y \mid Z)]$). This term measures the degree of inconsistency of Q in relation to σ by utilizing the conditional mutual information, as discussed in Section 2. This method is in contrast from the hard constraints used in the QCLP formulation Section 4.1. The hyperparameters λ and μ serve as regularization coefficients, adjusting for discrepancies from the marginals and the degree of inconsistency in the target distribution Q. The methodology for tuning these hyperparameters is discussed in Section 6.

Intuitively, the optimization problem aims to find a distribution Q that aligns closely with the empirical distribution P^D while being consistent with the imposed constraint. The relaxed OT distance serves as a measure of this alignment, and the objective is to minimize this distance, ensuring that Q is a faithful representation of P^D that simultaneously satisfies the constraint.

The inclusion of the CI constraint term makes our new formulation non-convex. We address this non-convexity with an alternating algorithm, FastOTClean. Before we detail FastOTClean in Algorithm 2, we describe its main idea. In this algorithm, we sequentially focus on either the

transport plan π or the resulting distribution Q, optimizing one while holding the other constant. Initially, we can set Q to a distribution that meets the CI constraint σ. With this fixed value, our objective becomes a convex function, which we solve using the Sinkhorn matrix scaling algorithm discussed in Section 2. When we alternate, our goal becomes minimizing the divergence between Q and $\pi(\mathbf{v}')$. In this stage, Q must also align with the CI constraint σ.

To address this problem, we adopt an alternating minimization strategy. Initiating with an initial guess for Q, the algorithm first determines the optimal transport plan $\pi(\mathbf{v}, \mathbf{v}')$ between P^D and Q through Sinkhorn iterations. In the subsequent iteration, a new Q is constructed based on the target distribution of π, denoted $\pi(\mathbf{v}')$. Specifically, this Q is identified to be proximate to $\pi(\mathbf{v}')$ based on the KL divergence while also ensuring it either approximately or strictly satisfies the independence constraint. In subsequent iterations, the transport plan is recalibrated with respect to the revised Q. Hence, the procedure can be viewed as a two-layered iterative process where the outer loop identifies a relaxed OT map, and the inner loop refines the target distribution of this map to enforce the constraint. The core intuition behind this approach is twofold. Firstly, the outer loop endeavors to determine a transport plan that maps the empirical distribution of data to a target distribution proximate to Q, influenced by the regularization coefficient; its primary objective is to minimize the transport cost. Conversely, the inner loop evaluates the target distribution derived from the outer mapping and formulates a distribution in close alignment with it, ensuring adherence to the constraint. In essence, while the outer loop emphasizes on minimizing the transportation cost, the inner loop focuses on enforcing independence constraints.

The inner loop of this alternating algorithm, which reconstructs Q based on $\pi(\mathbf{v}')$ to satisfy the CI constraint, can be interpreted as a rank-one non-negative matrix factorization (as highlighted in Capuchin [47]). Specifically, when dealing with conditional mutual information, the problem aligns with non-negative matrix factorization using the KL divergence objective, which is inherently non-convex but is typically addressed using alternating algorithms (for approximate enforcement of a CI constraint, one can use approximate matrix factorization techniques [22]). For a specific value $z \in \mathcal{Z}$, we aim to determine matrices $\mathbf{W}_z$ of size $d_X \times 1$ and $\mathbf{H}_z$ of size $d_Y \times 1$. These matrices represent the joint and conditional distributions $Q(X', Z' = z)$ and $Q(Y' \mid Z' = z)$. They are chosen to minimize the divergence $D_{\mathrm{KL}}(\pi(X', Y', Z' = z) \mid \mathbf{W}_z \cdot \mathbf{H}_z^T)$. While the D_{KL} is convex with respect to either $\mathbf{W}_z$ or $\mathbf{H}_z$, it is not jointly convex for the pair $(\mathbf{W}_z, \mathbf{H}_z)$. Established alternating methods, along with their associated update rules from the matrix factorization domain, such as those highlighted by Lee [36], can be employed. Starting with a random setup, these methods update $\mathbf{W}_z$ and $\mathbf{H}_z$ until they converge. The final matrices help us shape a new Q that satisfies the independence constraint.

We outline the algorithm to solve the optimization problem in (11), denoted by FastOTClean, in Algorithm 2. It begins by setting initial values for the vectors $\mathbf{p}$, $\mathbf{q}$, and the cost matrix $\mathbf{C}$ (see Lines 1 to 2). The vector $\mathbf{q}$ is set up to represent probabilities in a distribution satisfying σ, which serves as a first guess for the resulting distributions Q. The vectors $\mathbf{u}$ and $\mathbf{v}$, and the matrix $\mathbf{K}$ are then prepared for Sinkhorn iterations (Line 3). The Sinkhorn method find a plan π between our original $\mathbf{p}$ and the estimate $\mathbf{q}$ by updating $\mathbf{u}$ and $\mathbf{v}$ until they stabilize (Line 6). See Section 2 on checking convergence. After this, the algorithm computes the transport plan π (Line 7) and shifts its focus to reconstructing $\mathbf{q}$. The reconstruction step (Line 13) employed an alternating algorithm as described before to update $\mathbf{q}$.

4.2.1 Analysis of the algorithm. We prove that the algorithm converges. In Section 6, we empirically demonstrate this algorithm's inner workings and convergence properties. In Section 5, we propose efficient strategies to optimize this algorithm further.

Theorem 4.3. *For the optimization problem outlined in Equation* (11)*, Algorithm 2 converges.*

Proof. Algorithm 2 can be understood as an iterative optimization over one variable, either the transport plan π or the distribution Q, while holding the other variable constant. When Q is fixed, optimization concerning the transport plan is smooth, differentiable, and strictly convex, ensuring that the Sinkhorn iterations converge, as established by [23]. Conversely, with a fixed π, the inner problem breaks down into an objective function that remains strictly convex with respect to each matrix separately, and the adopted update rule ensures convergence to a stationary point, as elaborated in [28]. This approach mirrors the Coordinate Descent method, where the objective function is convex for each individual coordinate. As per [52][theorem 5.1], this process guarantees convergence to a coordinate-wise minimum of the objective function. □

Algorithm 2: FastOTClean: Fast Computation of Probabilistic Data Cleaner for Conditional Independence

Input: Database D, cost function c, and CI constraint $\sigma : X \perp\!\!\!\perp Y \mid Z$
Output: Transport plan (probabilistic data cleaner) π

$\mathbf{p} \coloneqq vector(P^D)$; $\mathbf{C} \coloneqq matrix(c)$;
Randomly initialize $\mathbf{q}$ ▷ An initial guess for Q
$\mathbf{u} \coloneqq \mathbb{1}_{d_X}$; $\mathbf{v} \coloneqq \mathbb{1}_{d_Y}$; $\mathbf{K} \coloneqq e^{-\frac{\mathbf{C}}{\rho}}$; ▷ Sinkhorn Initialization
while $\mathbf{q}$ *is not converged* **do** ▷ Sinkhorn iterations
  while $\mathbf{u}$ *and* $\mathbf{v}$ *are not converged* **do**
    $\mathbf{u} \coloneqq (\mathbf{p} \oslash (\mathbf{K} \cdot \mathbf{v}))^{\frac{\rho\lambda}{\rho\lambda+1}}$, $\mathbf{v} \coloneqq (\mathbf{q} \oslash (\mathbf{K} \cdot \mathbf{u}))^{\frac{\rho\lambda}{\rho\lambda+1}}$;
  $\pi = diag(\mathbf{u}) \cdot \mathbf{K} \cdot diag(\mathbf{v})$;
  for *each* $z \in \mathcal{Z}$ **do**
    Initialize $\mathbf{W}_z$, $\mathbf{H}_z$ randomly.
    while $\mathbf{W}_z$ *and* $\mathbf{H}_z$ *are not converged* **do**
      Update $\mathbf{W}_z$ to minimize $D_{\mathrm{KL}}(\pi(X', Y', Z' = z) \mid \mathbf{W}_z \cdot \mathbf{H}_z^T)$ with $\mathbf{H}_z$ fixed
      Update $\mathbf{H}_z$ to minimize $D_{\mathrm{KL}}(\pi(X', Y', Z' = z) \mid \mathbf{W}_z \cdot \mathbf{H}_z^T)$ with $\mathbf{W}_z$ fixed
  Construct $\mathbf{q}$ using $\mathbf{W}_z$s and $\mathbf{H}_z$s computed in the previous step
return π;

5 OPTIMIZATIONS

We applied several optimizations to improve FastOTClean that we briefly explain below and show their efficacy in Section 6.

Default Optimization. We applied two straightforward yet effective optimizations: 1) Confining the transport plan's size to restrict mass movement solely within D's active domain to $\mathcal{V}$, excluding movement to the entire support. We explained this in the context of QCLP while defining decision variables in Section 4.1. This restriction can be further narrowed down to allow mass movement within a more limited subset. 2) Rather than randomly initializing the target distribution Q in FastOTClean, we initiated it with a distribution satisfying the CI constraint by applying Non-negative Matrix Factorization (NMF) to the empirical distribution of D, which our results demonstrated to aid faster convergence.

Warm Starting Sinkhorn. Convergence of the Sinkhorn iteration is a significant bottleneck in FastOTClean. We observe that our alternating algorithm, while it changes Q in each iteration in which we fix the transport plan, only makes slight adjustments, implying that the transport plan should undergo minor changes in the next iteration. Therefore, instead of initializing the Sinkhorn

Dataset	#tuples	#attr.	avg. dom	init. CMI
Adult	48,842	14	5.42	0.18770
COMPAS	10,000	12	2.4	0.05484
Car	1,728	6	3.67	0.03617
Boston	506	14	4.5	0.05983

Table 2. Datasets characteristics

scaling factors $\mathbf{u}$ and $\mathbf{v}$ with vectors of ones, adopting a warm starting approach by initializing them with the $\mathbf{u}$ and $\mathbf{v}$ from the previous iteration can significantly accelerate convergence. Our evaluation results indicate that this is a highly effective idea.

Unsaturated CI Constraints. So far, we assumed that $\sigma : X \perp\!\!\!\perp Y \mid Z$ represents a saturated CI constraint, implying $\mathbf{V} = \{X, Y, Z\}$. However, in many real-world scenarios, especially with high dimensional data, CI constraints may not be saturated.

For unsaturated constraints, we split $\mathbf{V}$, the set of attributes in the database D, into two sets: $\mathbf{U} = \{X, Y, Z\}$ (the attributes in σ) and $\mathbf{W} = \mathbf{V} \setminus \mathbf{U}$ (those not in σ). A naive method is to compute a transport plan π of size $d_{\mathcal{V}}^2$, considering all attributes in $\mathbf{V}$, including $\mathbf{W}$. Adapting methods from Section 4 for this scenario is straightforward but computationally expensive with high-dimensional data.

A more efficient strategy is to run FastOTClean for the marginal distribution P_{U}^D instead of P^D. This results in a smaller transport plan π_s of size $d_{\mathcal{U}}^2$ compared to π. With π_s, we construct π as follows: $\pi(\mathbf{v}, \mathbf{v}') = 0$ if $\mathbf{w} \neq \mathbf{w}'$, and $\pi(\mathbf{v}, \mathbf{v}') = \pi_s(\mathbf{u}, \mathbf{u}')P_{\mathbf{W}|\mathbf{U}}(\mathbf{w} \mid \mathbf{u})$ otherwise. This ensures no additional transport cost for moving masses between different values of $\mathbf{W}$ as there is no mass moved for $\mathbf{w} \neq \mathbf{w}'$. Thus, the cost associated with π is the same as π_s, making it optimal if π_s is optimal. Note that this requires the cost function to satisfy some basic properties, such as the cost of $\mathbf{uw} \rightarrow \mathbf{u'w}$ being equal to the cost of $\mathbf{u} \rightarrow \mathbf{u}'$, which is satisfied by the Euclidean distance and other cost functions in our work. Additionally, the use of $P_{\mathbf{W}|\mathbf{U}}(\mathbf{w} \mid \mathbf{u})$ ensures that π satisfies the marginal constraint $P^D(\mathbf{v}) = \pi(\mathbf{v})$. The resulting distribution Q from π satisfies σ as its marginal is Q_{U} which is known to satisfy σ.

6 EXPERIMENTS

In our experimental evaluation of OTClean, we seek to answer the following research questions: **Q1** How does the end-to-end performance of OTClean in terms of algorithmic fairness compare with baseline approaches? (Section 6.2) **Q2** In data cleaning tasks related to CIs, how does the performance of OTClean compare with the baselines? (Section 6.3) **Q3** How effective is OTClean in determining optimal repairs? This encompasses evaluating its convergence behavior, runtime performance, and efficacy of the optimizations. (Section 6.5)

Datasets. We used four datasets. The Adult and COMPAS datasets highlight the fairness aspect of OTClean's application, while the datasets Car and Boston showcase the efficacy of OTClean in data cleaning tasks. Table 2 provides an overview of these datasets.

Adult [1]. In the Adult dataset, or "Census Income,", each entry captures details like age, work class, education level, marital status, occupation, relationship status, race, gender, weekly working hours, and country of origin. The dataset's main objective is to predict if an individual earns over $50K annually.

COMPAS [4]. The COMPAS dataset from the Broward County Sheriff's Office in Florida predicts the likelihood of an individual re-offending. Key attributes include age, gender, race, criminal history,

risk scores, charge degree, and jail history. COMPAS is essential for studies focusing on the fairness implications of predictive policing.

Car [3]. The Car Evaluation dataset evaluates cars based on attributes like buying price, maintenance cost, number of doors, person capacity, and safety. Cars are classified based on their overall condition into unacceptable, acceptable, good, or very good.

Boston [2]. The Boston Housing dataset provides insights into the housing market in Boston, Massachusetts. It covers attributes like crime rate, residential zoning, average room count, distance to employment centers, and median home value. It's frequently used for regression analysis in predicting housing prices.

Baselines. We use baselines that we briefly review here.

Algorithmic fairness. In the realm of algorithmic fairness, the objective is to guarantee that decision-making algorithms operate equitably, avoiding discrimination based on sensitive attributes like race or gender. While there are myriad definitions of fairness in the literature, this study primarily focuses on *interventional fairness*, as articulated in [47]. This particular notion underscores the importance of enforcing CI within data. Consider a sensitive attribute S. Without loss of generality, let's assume S is binary where $S = 1$ denotes the protected (or sensitive) group and $S = 0$ the unprotected group. Further, consider an ML model with output $\hat{Y}$ trained on a set of features $\mathbf{X}$. The notion of interventional fairness divides $\mathbf{X}$ into two sets: *admissible variables* $\mathbf{A}$ and *inadmissible variables* $\mathbf{N}$. Admissible variables are those where the effect of the sensitive attribute on the outcome, mediated by these variables, is considered fair. In [47], the extent to which an ML model deviates from this fairness standard is quantified using *the Ratio of Observational Discrimination (ROD)*, defined as:

$$\text{ROD} = \frac{1}{|dom(A)|} \sum_{a \in A} \frac{P(\hat{Y} = 1|S = 0, a)P(\hat{Y} = 0|S = 1, a)}{P(\hat{Y} = 0|S = 0, a)P(Y = 1|S = 1, a)}$$

A ROD value of 1 signals the absence of any bias and is in correspondence to the conditional independence ($\hat{Y} \perp\!\!\!\perp S \mid \mathbf{A}$). In this paper, we employ the logarithm of the ROD for our analyses. A logarithmic ROD value of 0 is indicative of the absence of discrimination, while progressively higher values of the log ROD signify increasing levels of bias. The approach detailed in [47] reduces the challenge of training a fair ML model to the task of enforcing a CI constraint on the training data. They introduced several methods in this context, which we adopt as baselines for our evaluations. Their methods fall into two categories: Methods based on matrix factorization and MaxSat methods. From the first category, the "Cap(MF)" factorizes each joint probability distribution of P^D for a fixed value of $\mathbf{Z}$ by minimizing Euclidean norm, while "Cap(IC)" does the factorization by using marginals of the initial distribution. They also propose a problem reduction of repairing w.r.t a CI constraint to solving a general CNF formula, and they solve it using their MaxSat method "Cap(MS)". We also included a naive baseline referred to as "Dropped," where the model is trained solely on admissible variables, which is sufficient for enforcing intervention fairness, as demonstrated in [47].

Data Cleaning. In our data cleaning evaluation, we assess the performance of OTClean and compare it with various imputation and data cleaning methods. We consider five baselines for handling missing values: 1) Most frequent (MF) fills missing values with the most frequent values within the attribute, 2) k-nearest neighbors (kNN) identifies the most frequent values among neighboring data points for imputation, 3) GAIN uses Generative Adversarial Networks [57], and 4) Hyperimputation is a method that integrates multiple imputation techniques, blending traditional iterative imputation with deep learning [31]. We selected kNN and MF as basic, widely-used baselines. We compared OTClean with GAIN since it is a leading imputation method and Hyperimpute since it is known for its ability to surpass various imputation techniques. We also use

two baselines in scenarios with attribute noise: 1) using the dirty dataset as a simple baseline, and 2) Baran [40] as an advanced data cleaning method that utilizes comprehensive context information, including the value, co-occurring values, and attribute type, to generate correction candidates with high precision.

6.1 Tuning OTClean

Cost function. We employ two cost functions in our experiments. The first function calculates the cost as the Euclidean distance between two records after normalizing their attributes by dividing them by their standard deviation. The second function utilizes a distance learned through MLKR (Metric Learning for Kernel Regression [54]), a supervised metric learning technique that minimizes the leave-one-out regression error. We chose MLKR because it is widely used for distance learning and designed explicitly for supervised tasks like those in our settings. We label the results from the first cost function as OTClean-C1, while the cost function using the learned distance is labeled as OTClean-C2.

Regularization Coeffients. Two tuning parameters of FastOTClean are λ and $\frac{1}{\rho}$. As λ and ρ grow, our formulation of OTClean gets closer to the OT distance, and FastOTClean gives better results. However, as their values grow, the cost of running FastOTClean increases due to slower convergence. To find parameter values that balance runtime and fast convergence, we perform a grid search for each dataset to tune OTClean. OTClean has another parameter, μ, that quantifies the dissatisfaction of the CI constraint.

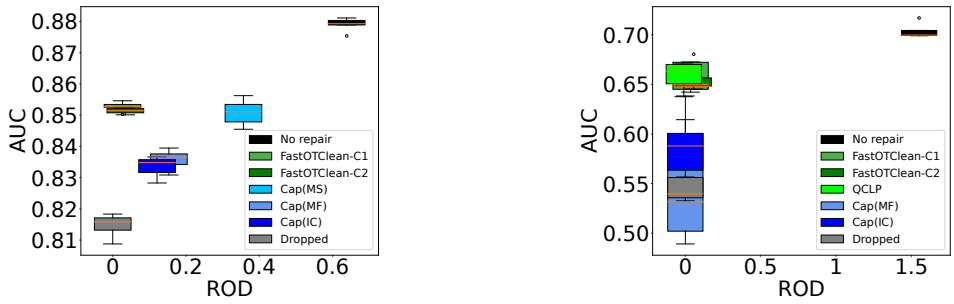
6.2 Algorithmic Fairness

We evaluate the effectiveness of OTClean within the domain of algorithmic fairness. To harness OTClean for training interventionally fair algorithms, we utilize our probabilistic data cleaning approach to modify the data, ensuring its consistency with the CI constraint ($S \perp\!\!\!\perp \mathbf{N} \mid \mathbf{A}$). This enforced independence ensures the sensitive attribute does not influence the inadmissible variable, except through $\mathbf{A}$. If this independence is maintained, any valuable predictive information encapsulated within the inadmissible variables $\mathbf{N}$ cannot be sourced from the sensitive attribute. The flexibility of our approach, underpinned by OT, allows us to craft specific cost functions for probabilistic data cleaning to preserve as much predictive capability as possible. Specifically, we designed a cost function to modify the inadmissible variables and keep sensitive attributes and admissible variables unchanged, ensuring that while fairness is achieved, all relevant predictive information within $\mathbf{A}$ is retained. Additionally, it ensures that any remaining predictive value within $\mathbf{N}$ is not derived from the sensitive attribute S.

We applied OTClean to establish a probabilistic data cleaner for the training data. This cleaner was subsequently used to pre-process the dataset. The subsequent sections present evaluation results on the `Adult` and `COMPAS` datasets. Our evaluation metrics include cross-validated AUC and the mean ROD averaged over iterations derived from cross-validation outcomes. Besides ROD, we also assess other fairness measures, such as equality of odds and demographic parity. Notably, our approach incidentally enhances these fairness metrics as well. We also report other popular fairness measures, such as equality of odds— which requires that classifiers have equal false positive and false negative rates across protected groups—and demographic parity, which ensures that the decision outcome is independent of the protected attribute.

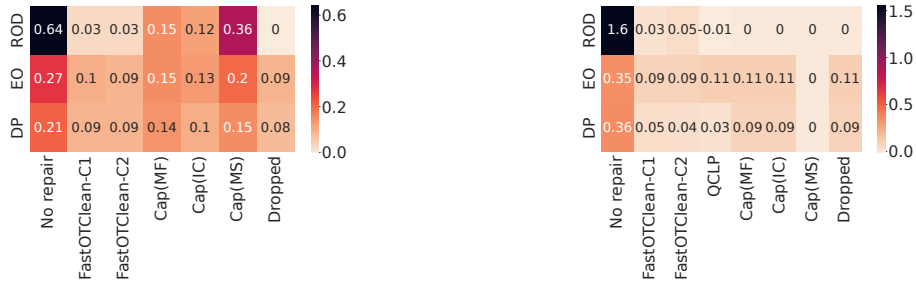
Figure 4 showcases our evaluation results for the `COMPAS` and `Adult` datasets. In the `Adult` dataset, the sensitive attribute is "sex", "marital-status" is inadmissible, and the admissible attributes include "occupation", "education-num", "hours-per-week", and "age". For `COMPAS`, we treat "race" as sensitive, "age-cat" and "priors-count" as inadmissible, and "charge-degree" as admissible. Notably, OTClean demonstrates superiority over the baseline, achieving models that are at least as fair, if not fairer, and exhibit an elevated AUC. This improvement can be attributed to our OT-based

approach, which empowers our method to retain considerable predictive value while rigorously enforcing fairness constraints. Furthermore, Figure 5 shows OTClean's reasonable performance on other fairness notions, specifically Equality of Opportunity (EO) and Demographic Parity (DP). On both datasets, our methodology consistently surpasses the baseline in these respects. (Note: the result of "Cap(MS)" is not plotted in Figure 4b as it achieved a constant AUC of 0.5 in all cross-validation iterations.)



a) Adult b) COMPAS

Fig. 4. Comparison of OTClean's performance with the baselines showing higher AUC and lower ROD (bias)



a) Adult b) COMPAS

Fig. 5. Fairness metrics in OTClean, indicating lower biases (ROD, EO, and DP) compared to baseline methods

6.3 Data Cleaning

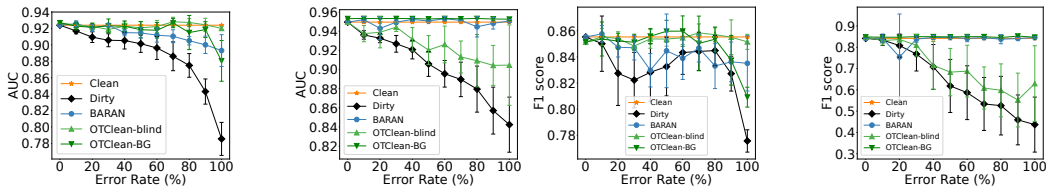
To evaluate the performance of OTClean in data cleaning, we conducted experiments using semi-synthetic datasets that featured two types of dirty data: *attribute noise* and *missing values*. These datasets were derived from the `Car` and `Boston` datasets. We used these datasets to train ML models for predicting the labels "class" (indicating the car's condition in the `Car` dataset) and "medv" (representing median house price in the `Boston` dataset), respectively. In each case, we introduced noise errors and missing values into the training data, while the original clean data served as the test set for assessing model generalization. For `Car`, we considered the CI constraint (doors $\perp\!\!\!\perp$ class | the remaining attributes). This constraint implies that the number of car doors should not significantly impact the class label when considering other factors such as buying price and safety. For the `Boston` dataset, we examined the constraint (B $\perp\!\!\!\perp$ medv | the remaining attributes), which suggests that the "B" attribute (indicating the percentage of blacks per town) should not influence the "medv" label. Initially, these constraints were

approximately held in the original datasets. To introduce attribute noise, we deliberately added non-random noise that led to violations of the CI constraints. Additionally, we injected two types of missingness: missing at random (MAR) and missing not at random (MNAR).

We chose to use a semi-synthetic dataset, where we added errors to real-world data, to create both "dirty" datasets and their accurate ground truths. This was essential because it is difficult to find real datasets with both genuine errors and ground truth. A limitation of this approach is that the injected error patterns may not exactly replicate those in actual datasets. However, our cleaning system is designed to be effective regardless of the specific error types. It primarily targets fixing spurious correlations and reducing the impact of any differences in error patterns on our goals.

To create a dependency between two attributes through attribute noise, we introduce random noise into one based on the values of the other. Our approach to adding missing data depends on the type. In MAR scenarios, where another attribute influences an attribute's missingness, we add missing values based on the other attribute's values in the same record. In MNAR cases, where an attribute's missingness is affected by its own value and other attributes, we randomly select records and determine missingness based on these factors. This method systematically creates relationships between attributes, effectively incorporating noise and addressing different missing data situations.

To assess the efficacy of OTClean, we utilized the "Dirty" datasets to train various ML models, including logistic regression, random forest, SVM, and MLP, and reported results for the best-performing model. When dealing with missing values, we employed two imputation methods: most frequent values (MF) and kNN, as explained previously. The dirty model is labeled with the imputation method used for training the dataset. In all experiments, the models were tested on ground truth data (the data before adding noise or missing values), and the models trained on the ground truth were denoted as "Clean." Additionally, we applied OTClean to enforce the corresponding CI constraint before training the ML models. This step aimed to remove spurious correlations induced by violations of CI, which could lead to poor performance of the ML model.

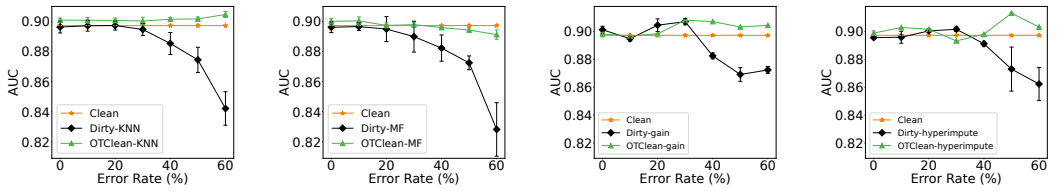


a) Car b) Boston c) Car d) Boston

Fig. 6. Attribute noise

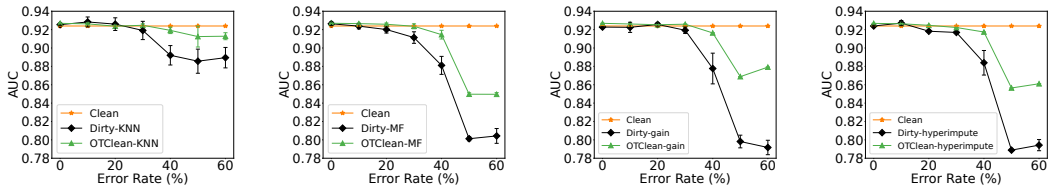
Attribute Noise. Figure 6 shows our results for cleaning data with attribute noise. We compared the performance, in terms of AUC and F1-score, of models using "Clean" data, "Dirty" data, and data cleaned by OTClean and Baran. Our cleaning algorithm only applies the CI constraint and does not need prior information about the noise type. However, it can also use knowledge about which attribute is noisy for repair. We tested OTClean in two ways: "blind", without knowing the noisy attribute, and with background knowledge (BG), where the noisy attribute is identified. The figures show how accuracy changes with different levels of noise. As noise increases, the model trained on dirty data performs worse. In contrast, the model trained on OTClean-cleaned data in both scenarios closely matches the ground truth model's behavior. This is because the dirty data model might learn false patterns not present in clean test data. However, using OTClean to apply the CI constraint helps the model focus on the correct data patterns. While OTClean improves

accuracy in both the blind and BG-informed settings, using background knowledge generally leads to better performance than the blind approach and Baran.



a) kNN Imputation b) Most Frequent c) GAIN d) Hyperimpute

Fig. 7. Missing at random (MAR) in Boston dataset



a) kNN Imputation b) Most Frequent c) GAIN d) Hyperimpute

Fig. 8. Missing Not at Random (MNAR) in Car dataset

Missing Values. In our missing value experiments (Figures 7 for MAR and 8 for MNAR), we tested model performance at different missing data levels. We compared "Dirty" models (trained with missing values filled using methods like MF, kNN, GAIN, and Hyperimpute) against OTClean-enhanced models (OTClean-MF, OTClean-KNN, OTClean-GAIN, and OTClean-Hyperimpute). For MAR, all imputation methods struggled with high missing data rates, affecting performance. However, combining them with OTClean improved results, closely matching the ground truth regardless of missing data amount. The slight advantage over ground truth models in Figure 7 is due to limited data size. For MNAR, as shown in Figure 8, our approach performed better than the baseline but declined as missing data increased. This is because MNAR issues are generally harder to address. While using OTClean helps reduce false correlations, differences in training and test data distributions can still affect performance.

6.4 Evaluation using Statistical Distortion

Dasu et al. [19] proposed a way to evaluate data cleaning methods, focusing on how they statistically distort data. They used measurements like the Earth Mover Distance (EMD) to see how much a method changes the original data distribution; less change is better. Their approach starts with a dirty dataset and its cleaned version. Using sampling, they generate pairs of these datasets, called replications, and clean the dirty ones. Using several replications instead of a single dataset pair ensures a more comprehensive and robust evaluation, avoiding biases that might arise from the unique characteristics of a single dataset. They then measure how much these strategies alter the data and improve error correction.

In our experiments, we applied this framework to test OTClean as a data cleaning method. We compared its effect on data distortion to other methods. Instead of looking at repaired errors, we focused on the accuracy (AUC) using the cleaned data. We ran 100 replications with attribute noise. The results are in Figure 9, where each cluster represents a cleaning method (the black point

shows the original dirty data). Each point shows the balance between data distortion and AUC improvement for replication. The figure indicates that OTClean generally improves performance more than Baran in most cases and is closer to the clean datasets, though with a bit more distortion. This increased distortion is due to moving the data closer to the ideal clean dataset, leading to better accuracy.

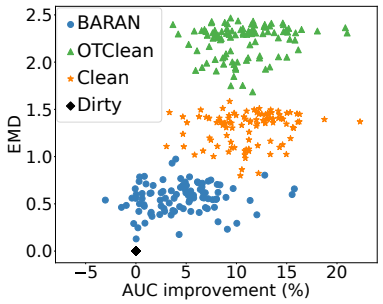


Fig. 9. Comparing OTClean and the competing cleaning methods based on their statistical distortion

Dataset	FastOTClean		MF	IC	MS	QCLP
	C1	C2				
Adult	4963	2158	66	66	700	NA
COMPAS	1137	846	7	6	1227	2

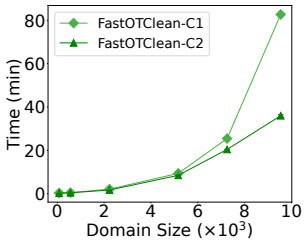
Table 3. Runtime (sec) for the fairness application

6.5 OTClean's Runtime and Performance

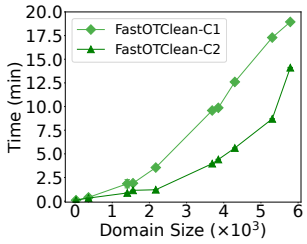
Runtime. In Table 3, we provide the runtime results of FastOTClean for Adult and COMPAS datasets, comparing them with the baselines. While our algorithm's runtime is somewhat higher due to the complex nature of OT, it remains reasonably fast and offers a practical means for employing OT in data cleaning for CI constraints. Our algorithm's runtime is mainly influenced by the number of attributes in the CI constraints rather than the data size. This is because the size of the transport plan we use stays the same no matter how large the data is; it only changes based on the number of attributes. In our experiments, the main factor we consider is the domain size, which depends on the number of attributes in the CI constraints and how many values these attributes can have. Figure 10 shows how FastOTClean's runtime and memory usage change with increasing domain size for the Adult and COMPAS datasets. Figures 10a,10b, and10c demonstrate that FastOTClean can handle large domain sizes efficiently. Figures 10a and 10b display the runtime for different cost functions, C1 and C2, showing that the choice of cost function affects the runtime. Figure 10c shows memory usage, which does not vary with the cost function. We can further reduce memory needs and speed up processing by using a sparse representation of the transport plan, since it is naturally sparse. We plan to explore this approach in our future work.

Convergence and Optimization. Figure 12 demonstrates the convergence behavior of our main FastOTClean, affirming the result presented in Theorem 4.3. It shows the monotonic decrease of the objective function, which represents the cost of the transport plan with the number of iterations. Additionally, the graph compares the convergence properties of FastOTClean with two different initializations: one with a random initialization of $\mathbf{q}$ and another using NMF. Notably, initializing with NMF reduces the total convergence iterations by nearly 30%. We also highlight optimizations aimed at reducing runtime. The first optimization involves updating q slices in parallel, achieving a significant speedup of $\times 7$ in our Adult data. Another optimization focuses on unsaturated CIs. Figure 11a illustrates the substantial runtime improvement achieved by employing the proposed optimization for unsaturated CI constraints while maintaining the same outcome. In this scenario, we initiate with a CI constraint and construct $\mathbf{W}$ using attributes with varying domain sizes. We then evaluate the runtime of both the naive and saturation approaches. The saturation approach

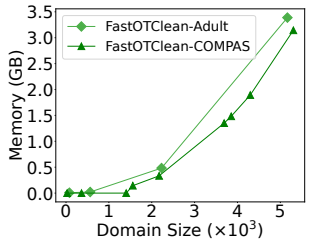
consistently solves the same problem, optimizing π_s, regardless of growing π's size, contributing to its stable performance. In our final experiment, we investigate the impact of warm start optimization on Sinkhorn iteration numbers. Figure 11b shows warm start reduces the number of iterations by more than sevenfold.



a) Runtime in Adult

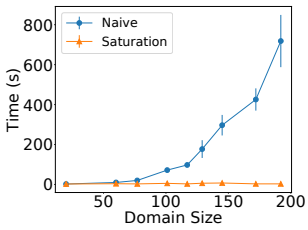


b) Runtime in COMPAS

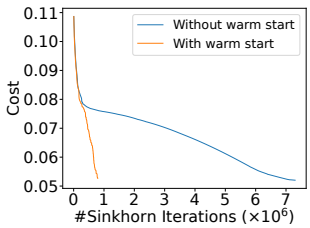


c) Memory usage in Adult and COMPAS

Fig. 10. OTClean's performance



a) Saturation



b) Warm start

Fig. 11. OTClean's Optimizations

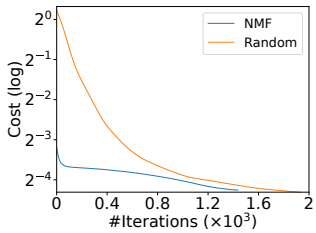


Fig. 12. Convergence

7 RELATED WORK

Our research connects with two main areas of study.

Data Cleaning. Data cleaning in the database domain traditionally revolves around enforcing integrity constraints, such as functional dependencies and conditional functional dependencies [10, 33, 37, 38]. Nonetheless, the domain of data cleaning for CI has only recently gained attention. Notable works in this emerging field include [47] and [56]. SCODED [56] employs statistical constraints to represent conditional dependence or independence and uses them to detect errors within datasets. However, it primarily focuses on ranking individual data tuples based on their relevance to CI violations, differing from our more general data-centric approach. The work in [47], on the other hand, aims to find optimal repairs for CI violations involving the addition or removal of tuples to satisfy the constraint. However, their method lacks the application of specific statistical divergence or distance measures to assess the quality of the repaired data. In a somewhat distinct vein, [5] utilizes generative adversarial networks (GANs) to generate data adhering to CI constraints. Their primary objective is to train these generative models effectively, particularly emphasizing the minimization of Jensen–Shannon divergence in continuous data. However, their focus is on training generative models rather than cleaning existing data.

Fairness and Optimal Transport. Algorithmic fairness research has primarily focused on detecting and mitigating biases in machine learning models, utilizing pre-, post-, and in-processing techniques. Pre-processing methods [15, 30] aims to eliminate bias from training data before model training. While model-agnostic approaches such as [14, 21, 47] exist, they often lack insights into the root causes of biases. These strategies typically address basic fairness criteria and may not delve into enforcing CI tests or incorporating OT methods. Notably, the authors in [49] employ OT as a regularizer during ML model training, focusing on a different aspect than our data cleaning objective. Additionally, studies like [9, 48] use OT to quantify unfairness, making them less aligned with our core research goal. The Wasserstein distance and barycenter are widely used in fair machine learning as solutions to the OT problem. For example, the study in [16] employs a fair classification approach that minimizes Wasserstein-1 distances to decouple classifier outputs from sensitive information, demonstrating strong empirical performance across various datasets. Similarly, the research in [25] uses the Wasserstein barycenter to preprocess training data to achieve statistical parity, though it does not delve into the complexities of conditional statistical inference in high-dimensional datasets, which sets it apart from our approach. Additionally, barycenters are utilized in [32] for learning real-valued functions that meet the Demographic Parity constraint. This work leverages OT theory to identify the optimal fair predictor, shown as the Wasserstein barycenter of distributions from sensitive groups. The paper also introduces a straightforward post-processing algorithm that effectively balances fairness with minimal increases in error rates, supported by empirical evidence.

8 CONCLUSION

In this paper, we formalize the problem of repairing and cleaning data with respect to conditional independence (CI) constraints using optimal transport theory. Optimal transport provides a mathematically rigorous framework for measuring the discrepancy between probability distributions, which is crucial for adjusting datasets while preserving their statistical properties. We developed an efficient algorithm that leverages approximations based on the Sinkhorn's matrix scaling technique, which is particularly suited for handling discrete data. This algorithm enables us to efficiently align data distributions with desired CI constraints without extensive computational costs typically associated with optimal transport solutions. Through experimental evaluation, we demonstrated that our approach not only adheres closely to CI constraints but also maintains the utility and accuracy of the data, surpassing the baseline methods in both performance and efficiency. As we look ahead, our research will expand OTClean to effectively manage the challenges associated with continuous data, which are prevalent in real-world applications. This expansion is critical for preserving the integrity and distribution of continuous datasets, ensuring that our data cleaning methods remain effective across different data types. Furthermore, we intend to explore the simultaneous enforcement of multiple CI constraints, a necessity in practical settings where data quality issues and biases are often intertwined and complex. This effort will involve developing robust data cleaning methods that can handle not only CI violations but also the intricacies of other database dependencies, such as functional and multivalued dependencies. Such dependencies are vital considerations in machine learning applications, where the accuracy and reliability of models heavily depend on the correct representation and distribution of the underlying data.

9 ACKNOWLEDGMENTS

This work has been partially supported by the National Science Foundation (NSF), grant OAC-2112606 and the National Institutes of Health (NIH), grant U54HG012510, The NSERC Discovery Launch Supplement DGECR-2021-00447 and NSERC Discovery Grants RGPIN-2021-04120.

REFERENCES

[1] 2023. Adult Data Set. https://archive.ics.uci.edu/ml/datasets/adult
[2] 2023. The Boston Housing Dataset. https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html
[3] 2023. Car Evaluation Data Set. https://archive.ics.uci.edu/ml/datasets/Car+Evaluation UCI Machine Learning Repository.
[4] 2023. COMPAS Analysis. https://github.com/propublica/compas-analysis/
[5] Kartik Ahuja, Prasanna Sattigeri, Karthikeyan Shanmugam, Dennis Wei, Karthikeyan Natesan Ramamurthy, and Murat Kocaoglu. 2021. Conditionally Independent Data Generation. In *UAI*. 2050–2060.
[6] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant Risk Minimization. *arXiv preprint arXiv:1907.02893* (2019).
[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein Generative Adversarial Networks. In *ICML*. 214–223.
[8] Leopoldo Bertossi. 2006. Consistent Query Answering in Databases. *ACM SIGMOD Record* 35, 2 (2006), 68–76.
[9] Emily Black, Samuel Yeom, and Matt Fredrikson. 2020. FlipTest: Fairness Testing via Optimal Transport. In *FAccT*. 111–121.
[10] Philip Bohannon, Wenfei Fan, Floris Geerts, Xibei Jia, and Anastasios Kementsietsidis. 2006. Conditional Functional Dependencies for Data Cleaning. In *ICDE*. 746–755.
[11] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.
[12] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. 2011. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine Learning* 3, 1 (2011), 1–122.
[13] Engin Bozdag. 2013. Bias in Algorithmic Filtering and Personalization. *Ethics and Information Technology* 15, 3 (2013), 209–227.
[14] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. 2017. Optimized Pre-Processing for Discrimination Prevention. *NeurIPS* 30 (2017).
[15] Simon Caton and Christian Haas. 2023. Fairness in Machine Learning: A Survey. *Computing Surveys* (2023).
[16] Evgenii Chzhen, Christophe Denis, Mohamed Hebiri, Luca Oneto, and Massimiliano Pontil. 2020. Fair regression with wasserstein barycenters. *Advances in Neural Information Processing Systems* 33 (2020), 7321–7331.
[17] Thomas M Cover. 1999. *Elements of Information Theory*. John Wiley & Sons.
[18] Marco Cuturi. 2013. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. *NeurIPS* 26 (2013).
[19] Tamraparni Dasu and Ji Meng Loh. 2012. Statistical Distortion: Consequences of Data Cleaning. *VLDB* 5, 11 (2012).
[20] Wenfei Fan and Floris Geerts. 2022. *Foundations of Data Quality Management*. Springer Nature.
[21] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In *KDD*. 259–268.
[22] Lorenzo Finesso and Peter Spreij. 2004. Approximate Nonnegative Matrix Factorization via Alternating Minimization. *arXiv preprint math/0402229* (2004).
[23] Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya, and Tomaso A Poggio. 2015. Learning with a Wasserstein Loss. *NeurIPS* 28 (2015).
[24] Sainyam Galhotra, Romila Pradhan, and Babak Salimi. 2021. Explaining Black-Box Algorithms using Probabilistic Contrastive Counterfactuals. In *SIGMOD*. 577–590.
[25] Paula Gordaliza, Eustasio Del Barrio, Gamboa Fabrice, and Jean-Michel Loubes. 2019. Obtaining Fairness using Optimal Transport Theory. In *ICML*. 2357–2365.
[26] Isabelle Guyon and André Elisseeff. 2003. An Introduction to Variable and Feature Selection. *JMLR* 3, Mar (2003), 1157–1182.
[27] Sara Hajian, Francesco Bonchi, and Carlos Castillo. 2016. Algorithmic Bias: From Discrimination Discovery to Fairness-Aware Data Mining. In *KDD*. 2125–2126.
[28] Le Thi Khanh Hien and Nicolas Gillis. 2021. Algorithms for Nonnegative Matrix Factorization with the Kullback–Leibler Divergence. *SISC* 87, 3 (2021), 1–32.
[29] Sara Hooker. 2021. Moving Beyond "Algorithmic Bias is a Data Problem". *Patterns* 2, 4 (2021), 100241.
[30] Maliha Tashfia Islam, Anna Fariha, Alexandra Meliou, and Babak Salimi. 2022. Through the data management lens: Experimental analysis and evaluation of fair classification. In *Proceedings of the 2022 International Conference on Management of Data*. 232–246.
[31] Daniel Jarrett, Bogdan C Cebere, Tennison Liu, Alicia Curth, and Mihaela van der Schaar. 2022. Hyperimpute: Generalized Iterative Imputation with Automatic Model Selection. In *ICML*. 9916–9937.
[32] Ray Jiang, Aldo Pacchiano, Tom Stepleton, Heinrich Jiang, and Silvia Chiappa. 2020. Wasserstein fair classification. In *Uncertainty in artificial intelligence*. PMLR, 862–872.

[33] Solmaz Kolahi and Laks VS Lakshmanan. 2009. On Approximating Optimum Repairs for Functional Dependency Violations. In *ICDT*. 53–62.
[34] Daphne Koller and Nir Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT press.
[35] Daphne Koller, Mehran Sahami, et al. 1996. Toward Optimal Feature Selection. In *ICML*, Vol. 96. 292.
[36] Daniel Lee and H Sebastian Seung. 2000. Algorithms for Non-Negative Matrix Factorization. *NeurIPS* 13 (2000).
[37] Ester Livshits and Benny Kimelfeld. 2022. The Shapley Value of Inconsistency Measures for Functional Dependencies. *LMCS* 18 (2022).
[38] Ester Livshits, Benny Kimelfeld, and Sudeepa Roy. 2020. Computing Optimal Repairs for Functional Dependencies. *TODS* 45, 1 (2020), 1–46.
[39] Sara Magliacane, Thijs Van Ommen, Tom Claassen, Stephan Bongers, Philip Versteeg, and Joris M Mooij. 2018. Domain Adaptation by Using Causal Inference to Predict Invariant Conditional Distributions. *NeurIPS* 31 (2018).
[40] Mohammad Mahdavi and Ziawasch Abedjan. 2020. Baran: Effective Error Correction via a Unified Context Representation and Transfer Learning. *VLDB* 13, 12 (2020), 1948–1961.
[41] Judea Pearl et al. 2009. Causal Inference in Statistics: An Overview. *Statistics Surveys* 3 (2009), 96–146.
[42] Ofir Pele and Michael Werman. 2009. Fast and Robust Earth Mover's Distances. In *ICCV*. 460–467.
[43] Roman Pogodin, Namrata Deka, Yazhe Li, Danica J Sutherland, Victor Veitch, and Arthur Gretton. 2023. Efficient Conditionally Invariant Representation Learning. *ICLR* (2023).
[44] Mateo Rojas-Carulla, Bernhard Schölkopf, Richard Turner, and Jonas Peters. 2018. Invariant Models for Causal Transfer Learning. *JMLR* 19, 1 (2018), 1309–1342.
[45] Babak Salimi, Bill Howe, and Dan Suciu. 2019. Data Management for Causal Algorithmic Fairness. *Data Engineering* (2019), 24.
[46] Babak Salimi, Bill Howe, and Dan Suciu. 2020. Database Repair Meets Algorithmic Fairness. *ACM SIGMOD Record* 49, 1 (2020), 34–41.
[47] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. 2019. Interventional Fairness: Causal Database Repair for Algorithmic Fairness. In *SIGMOD*. 793–810.
[48] Nian Si, Karthyek Murthy, Jose Blanchet, and Viet Anh Nguyen. 2021. Testing Group Fairness via Optimal Transport Projections. In *ICML*. 9649–9659.
[49] Chiappa Silvia, Jiang Ray, Stepleton Tom, Pacchiano Aldo, Jiang Heinrich, and Aslanides John. 2020. A General Approach to Fairness with Optimal Transport. In *AAAI*, Vol. 34. 3633–3640.
[50] Richard Sinkhorn. 1964. A relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices. *The Annals of Mathematical Statistics* 35, 2 (1964), 876–879.
[51] Antonio Torralba and Alexei A Efros. 2011. Unbiased Look at Dataset Bias. In *CVPR*. 1521–1528.
[52] Paul Tseng. 2001. Convergence of a Block Coordinate Descent Method for Nondifferentiable Minimization. *JOTA* 109, 3 (2001), 475.
[53] C Van de Panne. 1966. Programming with a Quadratic Constraint. *Management Science* 12, 11 (1966), 798–815.
[54] Kilian Q Weinberger and Gerald Tesauro. 2007. Metric Learning for Kernel Regression. In *AISTATS*. 612–619.
[55] SK Michael Wong, Cory J Butz, and Dan Wu. 2000. On the Implication Problem for Probabilistic Conditional Independency. *SMC* 30, 6 (2000), 785–805.
[56] Jing Nathan Yan, Oliver Schulte, MoHan Zhang, Jiannan Wang, and Reynold Cheng. 2020. SCODED: Statistical Constraint Oriented Data Error Detection. In *SIGMOD*. 845–860.
[57] Jinsung Yoon, James Jordon, and Mihaela Schaar. 2018. GAIN: Missing Data Imputation using Generative Adversarial Nets. In *ICML*. 5689–5698.

Received October 2023; revised January 2024; accepted February 2024