COMPUTING ALL OPTIMAL PARTIAL TRANSPORTS

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the classical version of the optimal partial transport problem. Let μ (with a mass of U) and ν (with a mass of S) be two mass distributions with $S \leq U$. For a parameter $\alpha \in [0, S]$, consider the minimum-cost transport plan σ_α that transports a mass of α from ν to μ. An *OT-profile* captures the behavior of the cost of σ_α as α varies from 0 to S. OT-profile has been used for studying mathematical properties of optimal partial transports (see Figalli (2010)). In this paper, we consider the question of computing the OT-profile. When μ and ν are discrete mass distributions, we show that the OT-profile is a piecewise-linear non-decreasing convex function. Let K be the complexity[1] of this function, we present an exact algorithm to compute the OT-profile in $\tilde{O}(n^2K)$ time. Given $\delta > 0$, we also show that the algorithm by (Lahn et al., NeurIPS 2019) can be used to δ-approximate the OT-profile of μ and ν with another piecewise-linear function in $O(n^2/\delta + n/\delta^2)$ time. The complexity of this approximation is just $O(1/\delta)$. An OT-profile is arguably more valuable than the OT-cost itself and can be used within applications. For instance, under a reasonable assumption of outliers, we prove that the first derivative of the OT-profile sees a noticable rise before any of the mass from outliers is transported. By using the OT-profile, we get an improved prediction accuracy for an outlier detection experiment as well as estimation of class priors within PU-Learning experiments, both of which are conducted on real-datasets.

1 INTRODUCTION

Given two discrete probability distributions ν (with a mass of $S = 1$) with the set A as the support and μ (with a mass of $U = 1$) with B as the support, where $|A| + |B| = n$, in the *optimal transport* problem, one wishes to compute the minimum cost plan to transport mass from ν to μ. When the mass $U \neq S$, the problem is called the *unbalanced optimal transport*. In the *partial optimal transport* problem, given a parameter $\alpha \in [0, S]$, one wishes to determine the α-optimal partial transport cost which is the minimum work required to transport a mass of α from ν to μ.

Owing to its strong statistical properties, the optimal transport cost Villani (2003); Peyré & Cuturi (2019) is considered to be an attractive dissimilarity metric between probability distributions, and has found numerous applications in areas involving GANs, image processing, Arjovsky et al. (2017); Liu et al. (2018); Balaji et al. (2020); Lin et al. (2021); Schmitz et al. (2018); Chen et al. (2019), variational inference Ambrogioni et al. (2018), econometrics Galichon (2016) and other areas of natural science Schiebinger et al. (2019); Sun et al. (2020) and applied mathematics Santambrogio (2015). Similarly the unbalanced and partial optimal transport has been used for various problems that arise in machine learning, including GAN training, image processing, outlier detection and Positive Unlabelled (PU-) learningYang & Uhler (2018); Bonneel & Coeurjolly (2019); Chapel et al. (2020); Mukherjee et al. (2021) .

The exact optimal transport cost and plan, including in the unbalanced case, can be computed in $O(n^3 \log n)$ time. For a fixed value of $\alpha \in [0, S]$, one can easily reduce the problem of computing an α-optimal partial transport cost to solving an unbalanced instance of the optimal transport problem in $O(n^3 \log n)$ time: Create a *catchment* node r in the support of μ with a additional mass of $(S - \alpha)$. Let $A' = A \cup \{r\}$ be the new support of μ. Note that the total mass of A' is $U + S - \alpha$. For every

[1]The combinatorial complexity of such a piecewise-linear function is simply the number of line segments it contains.

$b \in B$, add an edge (r, b) with a cost of 0. To find the α-optimal partial transport, one can simply solve the unbalanced optimal transport between A' and B using an exact solver in $O(n^3 \log n)$ time.

Several algorithms approximate the optimal transport cost in $O(n^2 \text{poly}\{1/\delta, \log n\})$ time. Cuturi Cuturi (2013) introduced the Sinkhorn algorithm to solve the entropic regularized optimal transport problem and showed that it can be used to approximate the optimal transport cost within an additive factor of δ in $\tilde{O}(n^2/\delta^2)$ time. Since then, there has been significant research on the design of additive approximation algorithms and several algorithms achieve an execution time of $\tilde{O}(n^2/\delta)$ Lahn et al. (2019); Jambulapati et al. (2019); Quanrud (2019). The state-of-the-art execution time for approximating the Optimal Transport is achieved by the combinatorial algorithm by Lahn, Mulchandani and Raghvendra (LMR-algorithm). Their algorithm is based on adapting a classical graph theory algorithm Gabow & Tarjan (1989) and runs in $O(n^2/\delta + n/\delta^2)$ time Lahn et al. (2019).

In this paper, we study the classical version of the optimal partial transport problem which we introduce next.

We are given μ and ν whose supports are the point sets A and B, respectively. Let $G(A \cup B, A \times B)$ be a complete bipartite graph with $A \cup B$ as the vertex set and $A \times B$ as its set of edges. For any $a \in A$ (resp. $b \in B$), we associate a mass of μ_a (resp. ν_b) such that the $U = \sum_{a \in A} \mu_a$ and $S = \sum_{b \in B} \nu_b \leq U$ [2]. We refer to each point $a \in A$ (resp. $b \in B$) to be a demand (resp. supply) point and assume μ_a (resp. ν_b) to be a positive rational number. For any pair of points $a \in A$ and $b \in B$, we are given a non-negative cost $c(a, b) \in \mathbb{R}_{\geq 0}$ bounded by 1. The cost of transporting a supply of mass β from b to a is $\beta c(a, b)$. A transport plan is a function $\sigma : A \times B \to \mathbb{R}_{\geq 0}$ that assigns a non-negative value to each edge of G indicating the quantity of supply transported along the edge. The transport plan σ is such that the total supplies transported into (resp. from) any demand (resp. supply) node $a \in A$ (resp. $b \in B$) is bounded by the demand ν_a(resp. supply μ_b) at a (resp. b), i.e., $\sum_{b \in B} \sigma(a, b) = \mu_a$ (resp. $\sum_{a \in A} \sigma(a, b) = \nu_b$). For any $\alpha \in [0, S]$, we say that any transport plan σ is an α-partial transport plan if it transports a mass of α from ν to μ, i.e., $\sum_{(a,b) \in A \times B} \sigma(a, b) = \alpha$. The cost of the partial transport plan denoted by $w(\sigma)$ is given by $\sum_{(a,b) \in A \times B} \sigma(a, b) c(a, b)$. In the α-optimal partial transport problem, we are interested in finding a minimum-cost α-partial transport plan which we denote by σ^*_α. We define the *OT-profile* to be a function $\omega : [0, S] \to \mathbb{R}_{\geq 0}$ that maps a value $\alpha \in [0, S]$ to the cost of $w(\sigma^*_\alpha)$ as α goes from 0 to S. For discrete distributions, we show that this function is convex and piecewise-linear. Let K denote the combinatorial complexity of ω. Since, OT profile is a piecewise-linear convex function, its first derivative is a non-decreasing step-function. We denote this step function as $D\omega$ where $D\omega(\alpha)$ denotes the first derivative of the OT-profile at α.

Next, we define the notion of an *α-approximate OT-profile*. We say that a function $\overline{\omega} : [0, S] \to \mathbb{R}_{\geq 0}$ δ-approximates the OT-profile, if, for every $\alpha \in [0, S]$, $\omega(\alpha) \leq \overline{\omega}(\alpha) \leq \omega(\alpha) + S\delta$. Recollect that, when μ and ν are probability distributions, $U = S = 1$ and we get $\omega(\alpha) \leq \overline{\omega}(\alpha) \leq \omega(\alpha) + \delta$. Therefore, $\overline{\omega}(\alpha)$ is an additive approximation of $\omega(\alpha)$ and the function $\overline{\omega}$ represents an additive approximations of all optimal partial transports.

The notion of OT-profile is not a new one. Figalli (2010) used the OT-profile to prove several useful mathematical properties of the optimal partial transports for the case where the distributions are continuous and $c(u, v) = \|u - v\|^2$. In this paper, we address the question of computing the OT-profile and present exact and approximation algorithms to do so. We also establish an interesting mathematical property of the first derivative of the OT-profile that can be used towards detecting outliers in the support of the distribution.

Our Contributions: In this paper, we design a new exact and an approximation algorithm to compute the OT-profile and show how they can be used to identify points from the outlier class and the inlier class within the support of a distribution. All our results apply for any arbitrary cost function $c(\cdot, \cdot)$.

- First, we give a simple combinatorial algorithm to compute the exact OT-profile ω and its first derivative $D\omega$. Our algorithm is similar in style to the classical Hungarian algorithm for the assignment problem. Moreover, if every partial optimal transport is unique, our algorithm constructs ω and $D\omega$ in $O(n^2 K)$ time; here K is the combinatorial complexity of the OT-profile. By making

[2] We consider the generalized case for arbitrary and potentially unbalanced case. For the case where μ and ν are probability distributions $U = 1$ and $S = 1$

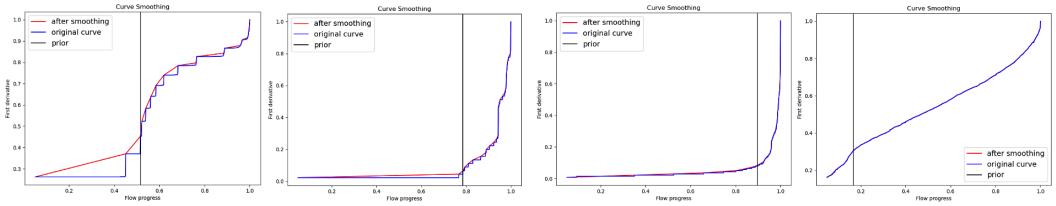

Figure 1: Examples of the first derivative of OT-profile from PU-learning Datasets, from left to right: mushroom, shuttle, pageblocks, usps

a slight modification to this algorithm, we are able to achieve an $\tilde{O}(n^2K)$[3] execution time without the uniqueness assumption. We show that ω is a piecewise-linear convex function and $D\omega$ is a non-decreasing step function.

- *Outlier Lemma:* Recollect A and B are supports of the distributions ν and μ respectively. Let A consists of points from the 'inlier' class (with a total mass of U) and B be a dataset that contains both points B^+ from the inlier class (with a total mass of α^*) and points B^- from the outlier class (with a total mass of $S - \alpha^*$). Let w denote the minimum-cost required to transport mass from B^+ to A. We assume that outlier points in B^- are separated from the inlier points of A as follows: *Assumption:* For some small $\varepsilon > 0$ and a constant $C > 1$, every outlier point $b \in B^-$ is at a distance at least Cw/ε from any inlier point in A. Under this assumption of the outlier points, we show the following lemma

  Lemma 1.1. *(A) The $(\alpha^* - \varepsilon)$-optimal partial transport generated by our algorithm will not transport any mass from the outlier points, and, (B) The first derivative values $D\omega(\alpha^* - \varepsilon) \leq w/\varepsilon$ and $D\omega(\alpha^* + \varepsilon) \geq Cw/\varepsilon$.*

  In other words, the first derivative curve will show a noticeable rise in an interval of width ε around α^*. By detecting this rise, we can approximate the values α^* and also mark the inliers and outliers.

- We show that the current state-of-the-art algorithm (LMR-algorithm) to approximate the OT cost can also generate a δ-approximation $\overline{\omega}$ of the OT-profile. The combinatorial complexity of $\overline{\omega}$ is only $\lceil 4/\delta \rceil$. Thus, one can not only approximate the OT-cost but construct an approximate OT-profile in $O(n^2/\delta + n/\delta^2)$ time. Moreover, we are also able to establish an approximate version of Lemma 1.1.

- We consider two applications where partial optimal transport has been successful and show how an OT-profile can help in improving accuracy while reducing the need for data dependent parameters. First, we conduct an outlier detection experiment that was proposed by Mukherjee et al. (2021). In this, we are given two sets of MNIST images, one set is the inlier (digits 0-4) the other is the mixture containing α^*-fraction of inliers (0-4) and $(1 - \alpha^*)$-fraction outlier (5-9) data. We can obtain α^* by identifying a sharp rise in the first derivative function $D\omega$. Then, we mark the free vertices with respect to α^*-optimal partial transport as outliers. In contrast, the method proposed in Mukherjee et al. (2021) depends on an additional data driven parameter that approximates the average distance between inliers. We conduct a second set of experiments for PU-learning that was proposed in Chapel et al. (2020). In this experiment, we are given a small set of positive samples and an unlabelled dataset. We are required to use them and identify (a) the fraction of positive samples in the unlabelled set (called the *class prior*), and, (b) classify all the samples in the unlabelled dataset as positive or negative. Chapel et al. (2020) presented a optimal partial transport based method that assumed the knowledge of class prior. In contrast, we use OT-profile to both identify the class prior α^* as well as generate a set of labels for the unlabelled data. In Figure 1, we show the first derivative of the OT-profile for four real-world PU-Learning experiments (See Section 5.2). In each case, one can observe a noticeable jump around the class prior. In both outlier detection as well as PU-Learning experiments, compared to previous work, we are able to achieve a better prediction accuracy as well as a reduced dependence on additional data dependent parameters.

[3]Here, $\tilde{O}(\cdot)$ hides poly $\log n$ factors.

2 PRELIMINARIES

In this section, we present notations and definitions that are required to describe the exact algorithm. Consider any transport plan σ transporting mass from ν to μ. We say that a vertex $a \in A$ (resp. $b \in B$) is *free* with respect σ if its demand (resp. supply) is not exhausted by σ, i.e., $\mu_a - \sum_{b \in B} \sigma(a,b) > 0$ (resp. $\nu_b - \sum_{a \in A} \sigma(a,b) > 0$). For free vertices, we refer to $\mu_a - \sum_{b \in B} \sigma(a,b)$ (resp. $\nu_b - \sum_{a \in A} \sigma(a,b)$) as the *deficit* (resp. *surplus*) at a (resp. b). At any stage in our algorithm, we use A_F (resp. B_F) to denote the set of free demand nodes (resp. free supply nodes). Recollect that $w(\sigma)$ is the cost of any transport plan σ with respect to $c(\cdot,\cdot)$.

We say that a transport plan σ is *dual feasible* if, for every $a \in A$ and $b \in B$, there are dual weights $y(a)$ and $y(b)$ associated with a and b respectively such that

$$y(a) + y(b) \leq c(a,b) \qquad \text{if } \sigma(a,b) < \min\{\mu_a, \nu_b\} \tag{1}$$

$$y(a) + y(b) \geq c(a,b) \qquad \text{if } \sigma(a,b) > 0. \tag{2}$$

These are the classical constraints corresponding to the dual formulation of the optimal transport problem. Let $y_{\max} = \max_{v \in A \cup B} |y(v)|$. In Lemma 2.1 (see Section 6.1 for a proof), we show that any feasible transport plan σ transporting a mass of α that satisfies the following property is an α-optimal partial transport plan:

(C) For every demand (resp. supply) node a (resp. b), the dual weight $y(a) \leq 0$ (resp. $y(b) \geq 0$) and, if a is a free demand (resp. free supply) node, then $y(a) = 0$ (resp. $y(b) = y_{\max}$).

To compute the OT-profile, therefore, we incrementally construct a transport plan while maintaining (C) as an invariant. By doing so, we guarantee that every intermediate transport plan is also an optimal partial transport plan. We can then use these intermediate transport plans to build the OT-profile.

Lemma 2.1. *Let σ along with dual weights $y(\cdot)$ be a feasible partial transport plan that transports a mass of α and satisfies (C). Then, σ is an α-optimal partial transport plan.*

Next, we introduce residual graphs that are extensively used in the design of combinatorial minimum-cost flow algorithms. We use $\overrightarrow{uv}$ to denote an edge directed from a vertex u to a vertex v.

Given a transport plan σ, one can build a *residual graph* $\overline{G}_\sigma$ as follows. The residual graph $\overline{G}_\sigma(A \cup B, \overrightarrow{E})$ is a directed graph with $A \cup B$ as the vertex set. For every $a \in A$ and $b \in B$,

- If $\min\{\mu_a, \nu_b\} > \sigma(a,b) \geq 0$, there is a *forward edge* $\overrightarrow{ba}$ to $\overrightarrow{E}$ and assign its residual capacity to be $\min\{\mu_a, \nu_b\} - \sigma(a,b)$.
- If $0 < \sigma(a,b) \leq \min\{\mu_a, \nu_b\}$, we add a *backward edge* $\overrightarrow{ab}$ to $\overrightarrow{E}$ with a residual capacity of $\sigma(a,b)$.

We would like to note that if $\sigma(a,b) = 0$ (resp. $\sigma(a,b) = \min\{\mu_a, \nu_b\}$), then there is only a forward (resp. backward) edge between a and b in $\overrightarrow{E}$. In all other cases, both the forward edge $\overrightarrow{ba}$ and the backward edge $\overrightarrow{ab}$ are in $\overrightarrow{E}$. We set the *cost* of any edge between a and b regardless of their direction to be $c(a,b)$. Given a partial transport plan σ, any directed path in the residual network $\overline{G}_\sigma$ that starts from a free supply vertex $b \in B_F$ and ends at a free demand vertex $a \in A_F$ is called an *augmenting path*. Note that, since the graph is bipartite, edges on any path alternate between a forward edge (directed from B to A) and a backward edge (directed from A to B). Any augmenting path P in $\overline{G}_\sigma$ can be used to increase the total mass that is transported along σ. We describe this process of *augmenting* σ along the path P next. For any augmenting path P from a free supply vertex b to a free demand vertex a, let s_b be the surplus at b and d_a be the deficit at a. We denote the bottleneck edge capacity κ of an augmenting path P to be the smallest residual capacity among all edges of P. Let $r_P = \min\{\kappa, s_b, d_a\}$ be the *bottleneck capacity* of the path P. We can augment $0 < k \leq r_P$ units of mass along an augmenting path P as follows. For every forward edge $\overrightarrow{ba}$ on the path P, we increase the mass transported from b to a to $\sigma(a,b) \leftarrow \sigma(a,b) + k$. For every backward edge $\overrightarrow{ab}$ on the path P, we reduce the mass transported from b to a to $\sigma(a,b) \leftarrow \sigma(a,b) - k$. We show that the updated σ continues to be a valid and feasible plan that transports an additional mass of k (See Lemma 2.2). Hence, we also refer to this operation as *pushing* a mass of k along P. Finally, we define the *net-cost*

of an augmenting path P as

$$\Phi(P) = \sum_{\overrightarrow{uv} \in P \text{ is forward}} c(u,v) - \sum_{\overrightarrow{uv} \in P \text{ is backward}} c(u,v).$$

By its definition, $\Phi(P)$ is simply the change in the cost of the transport plan when we push one unit of mass along P. Using this observation, we can show the following (see Section 6.1 for proof):

Lemma 2.2. *For any transport plan σ_α that transports a mass of α, let P be an augmenting path in the residual graph and let r_P be its residual capacity. For any $k \in [0, r_P]$, let $\sigma_{\alpha+k}$ be the transport plan obtained after pushing k supplies along P. Then $\sigma_{\alpha+k}$ is a valid transport plan with a cost $w(\sigma_{\alpha+k}) = w(\sigma_\alpha) + k\Phi(P)$.*

For any $(a,b) \in A \times B$, the *slack* of any forward edge $\overrightarrow{ba}$ in the residual network is

$$s(a,b) = c(a,b) - y(a) - y(b) \tag{3}$$

Any backward edge $\overrightarrow{ab}$ in the residual network has a slack of

$$s(a,b) = y(a) + y(b) - c(a,b) \tag{4}$$

For a feasible transport plan, all slacks are non-negative. Furthermore, note that if both $\overrightarrow{ab}$ and $\overrightarrow{ba}$ are present in the residual network, then the slack is 0. The next lemma, whose proof is in Section 6.1, relates the net-cost of any augmenting path to the slack of its edges.

Lemma 2.3. *Consider any augmenting path P with respect to a feasible transport plan σ. Suppose P starts at a free supply vertex $b \in B_F$ and ends at a free demand vertex $a \in A_F$. Then,*

$$\Phi(P) = y(b) - y(a) + \sum_{\overrightarrow{uv} \in P} s(u,v). \tag{5}$$

Next, we define any edge (a,b) in the residual graph $\overline{G}_\sigma$ as *admissible* if $s(a,b) = 0$. The *admissible graph* $\overline{\mathcal{A}}_\sigma$ is the subgraph of $\overline{G}_\sigma$ consisting of the admissible edges of the residual graph.

Lemma 2.4. *Consider a feasible transport plan σ that satisfies (C) and let P be an augmenting path consisting of admissible edges that starts at a surplus node $b \in B$ and ends at a deficit node $a \in A$. Then,*

$$\Phi(P) = y_{\max}. \tag{6}$$

Finally, given a residual network $\overline{G}_\sigma$, we define an *augmented residual network*, $\overline{\mathcal{G}}_\sigma(\mathcal{V}, \mathcal{E})$ as follows. In addition to $A \cup B$, the vertex set $\mathcal{V}$ also contains two additional special vertices s and t. The edge set $\mathcal{E}$ consists of two types of edges: (a) Every edge $\overrightarrow{uv} \in \overrightarrow{E}$ is also in $\mathcal{E}$. The *weight* of this edge is set to its slack $s(u,v)$, and (b) For every free supply (resp. demand) node $b \in B_F$ (resp. $a \in A_F$), $\mathcal{E}$ contains an edge $\overrightarrow{sb}$ (resp. $\overrightarrow{at}$) of weight 0. From the definition of slack, it follows that the weight of all edges in the augmented residual network in non-negative. The weight of any directed path in the augmented residual network is simply the sum of the weights of its edges.

3 Exact Algorithm

We begin by describing an algorithm to compute an optimal transport plan and establish a few of its properties. Then, we show how the intermediate computations in the execution of this algorithm can be used to compute the exact OT-profile.

Initialization Step: The algorithm picks an initial transport plan σ where, for every edge $(a,b) \in A \times B$, $\sigma(a,b) = 0$. Let α (initialized to 0) denote the mass transported by σ. The algorithm also initializes the dual weights of every vertex $v \in A \cup B$ to 0, i.e., $y(v) = 0$. Note that σ and $y(\cdot)$ together form a feasible transport plan.

Our algorithm executes in *phases*. Within each phase there are two *steps*.

First step (Hungarian Search): Execute Dijkstra's shortest-path algorithm on the augmented residual network $\mathcal{G}_\sigma$ with s as the source[4]. This algorithm computes the minimum weight path (or

[4]Recollect that s and t are special vertices added in the augmented residual network.

shortest path) from s to every vertex in the graph. For any vertex $v \in A \cup B$, let ℓ_v denote weight of the shortest path from s to v in $\mathcal{G}_\sigma$. The algorithm then uses these shortest path weights to update the dual weights of vertices. For any vertex $v \in A \cup B$, if $\ell_v \geq \ell_t$, the dual weight of v remains unchanged. Otherwise, if $\ell_v < \ell_t$, we update the dual weight as follows: **(U1):** If $v \in A$, we set $y(v) \leftarrow y(v) - \ell_t + \ell_v$, **(U2):** Otherwise, if $v \in B$, we set $y(v) \leftarrow y(v) + \ell_t - \ell_v$.

Second step: Let $\mathcal{A}$ be a graph obtained by adding an additional vertex s to the admissible graph, and, for every free supply vertex $b \in B_F$ adding an edge (s, b) directed from s to b. Starting from s, we execute a Depth First Search (DFS) in $\mathcal{A}$ to find a path to a free demand vertex. Let P' be a path returned by DFS that starts at s and ends at a free demand vertex $a \in A_F$. Let P be the path that remains after we remove the vertex s from P'. Note that P is a directed path in the residual graph starting at a free supply node and ending at a free demand node, i.e., P is an augmenting path. Let $r_P > 0$ be the bottleneck capacity of P. We update σ by augmenting a mass of r_P along the path P. After augmentation, we update $\alpha \leftarrow \alpha + r_P$.

The algorithm iteratively executes phases until σ becomes a maximum transport plan (i.e., $\alpha = S$). This completes the description of the algorithm.

The dual updates at the end of the first step of each phase guarantee that the transport plan σ along with the updated dual weights remains feasible and there is at least one augmenting path in the admissible graph (See Lemma 6.2 and 6.3 in Appendix for a proof). The second step computes an augmenting path P in the admissible graph and increments the mass transported by augmenting the transport plan along P. Using this and the fact that the demands and supplies are rational numbers, we show, in Section 6.2.1 that the algorithm terminates in finite number of phases with a maximum transport plan. Furthermore, we show that our algorithm maintains (C) as an invariant.

Lemma 3.1. *The algorithm terminates after executing a finite number of phases. Furthermore, (C) holds for the duration of the algorithm's execution.*

Before we describe the construction of the OT-profile, we introduce a few notations. Suppose the algorithm described above executes for q phases. Let $\{\sigma_0, \sigma_1, \ldots, \sigma_q\}$ be the partial transport plan maintained by the algorithm where σ_0 is the initial empty transport plan and σ_i is the transport plan at the end of phase i. Let $\{\alpha_0 = 0, \alpha_1, \ldots, \alpha_q = S\}$ be such that α_i is the mass transported by σ_i. For any $v \in A \cup B$, let $y_i(v)$ denote the dual weight of v after phase i and let $y^i_{\max} = \max_{v \in A \cup B} |y_i(v)|$. For $1 \leq i \leq q$, let P_i denote the augmenting path computed in phase i and r_i denote its bottleneck capacity. The following lemma, whose proof is in Section 6.2.2, will be useful in describing the construction of OT-profile and its first derivative.

Lemma 3.2. *For every $1 \leq i \leq q$, we have: (a) $w(\sigma_i) = \sum_{j=1}^{i} r_j y^j_{\max}$, and, (b) $y^i_{\max} = y^{i-1}_{\max} + \ell_t$.*

Generating the OT-profile ω and the first derivative $D\omega$: To generate the OT-profile ω, we set p_i to be a point $(\alpha_i, \sum_{j=1}^{i} r_j y^j_{\max})$ and set ω to be the piecewise-linear function given by the sequence of points $\langle p_0, p_1, \ldots, p_q \rangle$ where each consecutive pair in the sequence is connected by a line segment. We set the first derivative function $D\omega$ as follows: For $\alpha = 0$, $D\omega(\alpha) = y^1_{\max}$. For any $\alpha \in (\alpha_{i-1}, \alpha_i]$, we set $D\omega(\alpha) = y^i_{\max}$.

Correctness of the OT Profile: Next, we discuss why ω returned by the algorithm is indeed the OT-profile. Observe that, from Lemma 2.1 and Lemma 3.1, σ_i is an α_i-optimal partial transport and from Lemma 3.2(a), $\omega(\alpha_i) = w(\sigma_i) = \sum_{j=1}^{i} r_j y^j_{\max}$. Therefore, $p_0, p_1, \ldots, p_q$ are points on the OT-profile ω.

Lemma 3.3. *For any $0 < i \leq q$ and any intermediate value $\alpha \in (\alpha_{i-1}, \alpha_i)$, an α-optimal partial transport plan can be obtained by augmenting σ_{i-1} by a mass of $(\alpha - \alpha_{i-1})$ along the augmenting path P_i. The resulting transport plan has a cost of $\sum_{j=1}^{i-1} r_j y^j_{\max} + (\alpha - \alpha_{i-1}) y^i_{\max}$.*

From Lemma 3.3 (proof in Section 6.2.3), the point $p_\alpha = (\alpha, \sum_{j-1}^{i-1} r_j y^j_{\max} + (\alpha - \alpha_{i-1}) y^i_{\max})$ will be on the OT-profile. The set of points p_α as α goes from α_{i-1} to α_i will be the line segment connecting $p_{i-1} = (\alpha_{i-1}, \sum_{j=1}^{i-1} r_j y^j_{\max})$ and $p_i = (\alpha_i, \sum_{j=1}^{i} r_j y^j_{\max})$. Note that the slope of this line segment is precisely $y^i_{\max}$. Therefore, we can conclude the following:

Lemma 3.4. *For any $\alpha \in (\alpha_{i-1}, \alpha_i]$, $D\omega(\alpha) = y^i_{\max}$.*

From Lemma 3.2(b), $\langle y^1_{\max}, \ldots, y^q_{\max}\rangle$ is a non-decreasing sequence, ω is a non-decreasing piecewise-linear convex function.

Efficiency analysis: Each phase of our algorithm requires executing a single Dijkstra's shortest-path algorithm which takes $O(n^2)$ time. Therefore, the total time taken by the algorithm is $O(qn^2)$. Note that the OT-profile generated by our algorithm also has q points, i.e., $\omega = \langle p_0, p_1, p_2, \ldots, p_q\rangle$. However, if a subsequence $\langle p_i, p_{i+1}, \ldots, p_j\rangle$ are all collinear, one can remove the points that lie between p_i and p_j to obtain a sparser representation of ω. Thus, the number of phases may be higher than the complexity K of ω.

Suppose, for every value of $\alpha \in [0, S]$, we assume that the optimal partial transport is unique. In this case, we show (in the proof of Lemma 3.5 presented in Section 6.2.4) that $\langle y^0_{\max}, \ldots y^q_{\max}\rangle$ is a strictly increasing sequence and therefore no three points on the OT-profile generated by our algorithm are collinear, implying $q = K$.

Lemma 3.5. *Suppose, for every value of $\alpha \in [0, S]$, the optimal partial transport is unique, then the algorithm described above executes in $O(n^2K)$ time where K is the complexity of the OT-profile.*

In order to avoid collinear points, one can modify the second step of the algorithm as follows. In the second step of any phase i, instead of computing a single augmenting path in the admissible graph, we can compute a maximum flow by using a $\tilde{O}(n^2)$ time algorithm Chen et al. (2022). By doing so, we can guarantee that at the end of the phase i, there are no more augmenting paths in the admissible graph. Thus, Step 1 of phase $i + 1$ will adjust the dual weights and create an augmenting path in the admissible graph. Therefore, ℓ_t as computed in Step 1 of phase $i + 1$ is positive and from Lemma 3.2(b), we have $y^{i+1}_{\max} > y^i_{\max}$. $y^i_{\max}$ and $y^{i+1}_{\max}$ are slopes of line segments (p_{i-1}, p_i) and (p_i, p_{i+1}) respectively. Therefore, we conclude that no three points p_{i-1}, p_i, and p_{i+1} are collinear points in the OT-profile and so the number of phase q is exactly equal to the complexity K of the OT-profile.

Theorem 3.6. *The OT-profile of two mass distributions μ and ν can be computed in $\tilde{O}(n^2K)$ time.*

Now, we are ready to present the proof of Lemma 1.1. As part of our proof, we will assume that we can *generate*, for any $\alpha \in [0, S]$, an α-optimal partial transport along with a set of dual weights that satisfy (C) by using our exact algorithm as follows: If $\alpha \in \{\alpha_0, \alpha_1, \ldots, \alpha_q\}$, then return the explicitly constructed α-optimal partial transport plan and a set of dual weights that satisfy (C). For any other $\alpha \in (\alpha_{i-1}, \alpha_i)$, push a mass of $(\alpha - \alpha_i)$ along P_i and return the resulting transport plan and dual weights. As shown in Lemma 3.3, this is indeed an α-optimal partial transport plan which together with the dual weights satisfy (C).

4 Approximation Algorithm

Next, we show how a single execution of the LMR-algorithm (that has an execution time of $O(n^2/\delta + n/\delta^2)$ can be used to compute a δ-approximate OT-profile with a combinatorial complexity of $O(1/\delta)$. The LMR-algorithm is split into three parts. In the first part, the algorithm will scale-and-round the demands and supplies to integers. In the second part, the algorithm finds an approximate transport plan using a primal-dual method that is similar in style to the exact algorithm described in Section 3. Like in the exact case, the intermediate computations of this part can be used to compute an approximate OT-profile. Finally, in the third part, the transport plan is mapped back to the original demands and supplies.

The second part of the LMR-algorithm maintains a partial transport plan σ (initialized to 0) and dual weights $y(\cdot)$ (initialized to 0) that satisfy a relaxed set of feasibility conditions. This part of the algorithm executes in *phases* and terminates when σ becomes a maximum transport plan. Within each phase there are two *steps*. Similar to the exact algorithm, in the first step, the algorithm conducts a Hungarian Search and adjusts the dual weights so that there is at least one augmenting path of admissible edges. In the second step, the algorithm computes many augmenting paths and updates σ by augmenting it along all paths computed. At the end of the second step, we guarantee that there is no augmenting path of admissible edges. Lahn *et al.* Lahn et al. (2019) bounded the number of phases to $q = O(1/\delta)$ and the total execution time of the algorithm by $O(n^2/\delta + n/\delta^2)$. Let σ_0 be the initial transport plan and σ_i be the partial transport plan maintained after phase i. Let $\alpha_0 = 0$ and α_i be the mass transported by σ_i and let $y^i_{\max}$ be the largest dual weight at the end of phase i.

To approximate the OT-profile, we simply add, for each $0 \leq i \leq q$ the points $p_i = (\alpha_i, w(\sigma_i))$. The approximate OT-profile $\overline{\omega}$ is given by the piece-wise linear function $\langle p_0, p_1, \ldots, p_q \rangle$ where every consecutive pair of points in this sequence is connected by a line segment. Unlike in the exact case, $\overline{\omega}$ is not necessarily convex and its first derivative can be unstable. Nonetheless, we are able to define a step function $D\overline{\omega}$ that satisfies an approximate version of the outlier lemma: $D\overline{\omega}(0) = 0$ and for every $\alpha \in (\alpha_i, \alpha_{i+1}]$, $D\overline{\omega}(\alpha) = y^i_{\max}$. Details of the LMR-algorithm, the approximate version of the OT-profile as well as the approximate outlier lemma are presented in the Appendix (see 7, 7.2.1).

5 EXPERIMENTS

Experimental Setup: We conduct experiments for outlier detection as well as PU-Learning experiments on synthetic and real world dataset. Our algorithm is implemented in Java and experiments were conducted using Python, and are executed on a machine with 2.1 GHz Intel® Xeon® E5-2683v4 processor with 64 GB of RAM.

5.1 OUTLIER DETECTION

In this section, we test the effectiveness of our outlier detection lemma on a real-world dataset. Additional experiments on synthetic datasets are presented in the Appendix 8.4. Our lemma helps identify the inlier mass α^* based on an observable increase in the first derivative function around α^*.

Experiment with Real Data-set: In this experiment, we follow the setup of Mukherjee et al. (2021). Denote μ as the set of clean data containing n MNIST digits LeCun (1998) from 0 to 4, and ν as a mixed set contaminated by digits from 5 to 9. We set ε as the mixing ratio of outlier data in the whole dataset, i.e., $(1-\varepsilon)n$ digits in ν are between 0 and 4 and remaining εn fraction are outliers that are between 5 and 9. We utilize the L_1 distance (Manhattan distance) as the metric of MNIST digits. We execute two sets of experiments, one with $n = 1k$ and another with $n = 10k$ images. For each set, we consider $\varepsilon = 0.2, 0.25$ and 0.3. Our algorithm computes the OT-profile ω and its first derivative $D\omega$. Then, it uses the kneedle method (with a default sensitivity of 1) to catch a sudden rise in the first derivative $D\omega$. Thus, from the outlier lemma, we are able to estimate the value of ε with $\hat{\varepsilon}$ (we refer to this as the check point). We then label the images as inliers (resp. outliers) if they are matched (resp. free) with respect to the $(1-\hat{\varepsilon})$-optimal partial transport. See Appendix 8.2 for details regarding the application of the kneedle method. We repeat the experiments 30 times and report the average and standard deviation of the accuracy. We also conduct the same set of experiments for the robust optimal transport algorithm of Mukherjee et al. (2021). The results are presented in Table 1.

Table 1: Outlier Detection Accuracy(left) and Check Point Detected(right) for MNIST

Method(n)	$\varepsilon = 0.3$	$\varepsilon = 0.25$	$\varepsilon = 0.2$
ROBOT($1k$)	0.73	0.77	0.81
OT-Profile($1k$)	0.81	0.82	0.85
ROBOT($10k$)	0.77	0.80	0.839
OT-Profile($10k$)	0.88	0.89	0.90

Method(n)	$\varepsilon = 0.3$	$\varepsilon = 0.25$	$\varepsilon = 0.2$
ROBOT($1k$)	0.728	0.744	0.792
ROBOT($10k$)	0.754	0.770	0.796

Our outlier detection method consistently achieves a slightly better accuracy (between $4\% - 10\%$). Table 1 shows the check point that is automatically detected by our algorithm. Observe that the check point detected by our algorithm is very close to value of ε (the actual outlier ratio). Note that the method of Mukherjee et al. (2021) relies on a hyper-parameter that is obtained by sampling μ. Our method on the other hand, does not rely on any data driven hyperparameters. In the supplemental material, we show a visual sample of the miss-classified MNIST images for ROBOT and OT-profile approaches. We also provide number of errors made on a sample of $n = 2000$ by ROBOT and OT-Profile for each digit.

5.2 PU LEARNING

In this section, we apply OT-profile in the context of PU Learning Bekker & Davis (2020). The goal of PU learning is similar to binary classification problem where labelled positive examples (let n_P denote the number of positive examples) make up the training set where as the test set consists of positive and negative examples that are unlabelled (let n_U denote the number of unlabelled examples). In case of multiple classes, similar approach is followed where the target classes are considered as positive ($y = 1$) and all the others are considered as negative ($y = -1$). Following the previous work set up, we applied our approach under the SCAR assumption (selected completely at random) on six datasets from UCI repository, which are: Mushrooms, Shuttle, Pageblocks, USPS, Connect-4 and Spambase and test SAR (selected at random) assumption with colored MNIST setup. From each dataset, we randomly choose equal number of samples, $n_P = n_U = 800$ for the positive set as well as the unlabelled set. The experiments are run 10 times, and the mean accuracy and running times is reported.

Recently, Chapel et.al.Chapel et al. (2020) demonstrated that specific optimal partial transport formulations (namely Partial Wasserstein (p-W) and Partial Gromov-Wasserstein (p-GW)) can be used for learning positive unlabelled data. In Chapel et al. (2020), it is assumed that the true proportion of positives in the unlabelled dataset (denoted by $\pi = p(y = 1)$), called *class prior* is known in advance for different datasets. π is used to sample the test (unlabelled) set as per the marginal $p(x) = \pi p(x|y = 1) + (1 - \pi)p(x|y = -1)$. Both the methods, p-W and p-GW, need the knowledge of π.

On the other hand, we describe an approach that uses the OT-profile ω and the first derivative $D\omega$ to solve the PU-learning task without the knowledge of class prior. We refer to this approach as *OTP-wo-prior*. Since the unlabelled data can have samples from multiple classes, there can be several noticable rise in the first derivative $D\omega$. In such a case, we pick the earliest jump to estimate the class prior. Let $\hat{\pi}$ be the class prior estimate returned by our algorithm. We then mark all free points with respect to $\hat{\pi}$-optimal partial transport as negative and all other points are positive. We describe our method to detect a jump in $D\omega$ in the Appendix 8.3.

We also describe a method assuming the class prior π is known. If π is known, we simply mark the free points with respect to the π-partial transport plan as negative and the remaining points as positive. We refer to this approach as *OTP-w-prior*.

We observed that OTP-wo-prior is both computationally efficient as well as achieves comparable or better accuracy (see Table 2, standard deviation reported in Appendix 8) than the methods p-W and p-GW proposed by Chapel et al. (2020). Moreover, unlike previous methods, our method works does not need the knowledge of class prior which arguably is a significant limitation of the work by Chapel et al. (2020). Under SCAR assumption, except for usps and spambase, both OTP-w-prior(italic) and OTP-wo-prior(bold) outperform p-W and p-GW. And detected priors are close to the real value within 0.2% for mushrooms, shuttle, and pageblocks. Under SAR, OTP-w-prior yields the best accuracy, and OTP-wo-prior has similar result with p-W and P-GW.

Table 2: PU learning accuracy rates(left) and execution time(right) in seconds

dataset	π	p-W	p-GW	OTP-w-prior	OTP-wo-prior	$\hat{\pi}$	p-W	p-GW	OTP-w-prior	OTP-wo-prior
mushrooms	0.518	96.1	95.2	*99.8*	**99.7**	0.517	1.2	83.9	0.6	0.5
shuttle	0.786	96.3	95.5	*97.9*	**97.7**	0.785	1.0	82.3	0.5	0.4
pageblocks	0.898	92.4	90.6	*93.1*	**93.1**	0.925	1.4	92.1	0.3	0.3
usps	0.167	**98.6**	95.7	97.3	92.5	0.092	2.1	80.2	8.1	7.5
connect-4	0.658	61.0	55.4	*74.7*	**67.3**	0.860	2.2	82.0	1.9	1.8
spambase	0.394	**79.8**	71.1	73.4	65.9	0.301	1.2	80.3	0.8	0.7
mnist	0.1	99.1	98.4	*99.3*	98.1	0.081	1.8	82.3	3.1	2.6
colored mnist	0.1	91.6	97.5	*99.3*	**97.6**	0.076	1.9	81.8	3.2	2.9

REFERENCES

Luca Ambrogioni, Umut Güçlü, Yağmur Güçlütürk, Max Hinne, Marcel A van Gerven, and Eric Maris. Wasserstein variational inference. *Advances in Neural Information Processing Systems*, 31, 2018.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR, 2017.

Yogesh Balaji, Rama Chellappa, and Soheil Feizi. Robust optimal transport with applications in generative modeling and domain adaptation. *Advances in Neural Information Processing Systems Foundation (NeurIPS)*, 2020.

Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: A survey. *Machine Learning*, 109(4):719–760, 2020.

Nicolas Bonneel and David Coeurjolly. Spot: sliced partial optimal transport. *ACM Transactions on Graphics (TOG)*, 38(4):1–13, 2019.

Laetitia Chapel, Mokhtar Z Alaya, and Gilles Gasso. Partial optimal tranport with applications on positive-unlabeled learning. *Advances in Neural Information Processing Systems*, 33:2903–2913, 2020.

Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. *CoRR*, abs/2203.00671, 2022. doi: 10.48550/arXiv.2203.00671. URL https://doi.org/10.48550/arXiv.2203.00671.

Liqun Chen, Yizhe Zhang, Ruiyi Zhang, Chenyang Tao, Zhe Gan, Haichao Zhang, Bai Li, Dinghan Shen, Changyou Chen, and Lawrence Carin. Improving sequence-to-sequence learning via optimal transport. *arXiv preprint arXiv:1901.06283*, 2019.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013.

Alessio Figalli. The optimal partial transport problem. *Archive for rational mechanics and analysis*, 195(2):533–560, 2010.

H. N. Gabow and R.E. Tarjan. Faster scaling algorithms for network problems. *SIAM J. Comput.*, 18: 1013–1036, October 1989.

A. Galichon. *Optimal Transport Methods in Economics*. Princeton University Press, 2016.

Arun Jambulapati, Aaron Sidford, and Kevin Tian. A direct tilde {O}(1/epsilon) iteration parallel algorithm for optimal transport. *Advances in Neural Information Processing Systems*, 32, 2019.

Nathaniel Lahn, Deepika Mulchandani, and Sharath Raghvendra. A graph theoretic additive approximation of optimal transport. *Advances in Neural Information Processing Systems*, 32:13836–13846, 2019.

Yann LeCun. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. License: Creative Commons Attribution-Share Alike 3.0.

Hsin-Yi Lin, Huan-Hsin Tseng, Xugang Lu, and Yu Tsao. Unsupervised noise adaptive speech enhancement by discriminator-constrained optimal transport. *Advances in Neural Information Processing Systems*, 34, 2021.

Huidong Liu, Xianfeng Gu, and Dimitris Samaras. A two-step computation of the exact GAN Wasserstein distance. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3159–3168. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/liu18d.html.

Debarghya Mukherjee, Aritra Guha, Justin M Solomon, Yuekai Sun, and Mikhail Yurochkin. Outlier-robust optimal transport. In *International Conference on Machine Learning*, pp. 7850–7860. PMLR, 2021.

G. Peyré and Marco Cuturi. *Computational Optimal Transport: With Applications to Data Science*. Now Publishers Inc, 2019.

Kent Quanrud. Approximating optimal transport with linear programs. In *2nd Symposium on Simplicity in Algorithms*, pp. 1, 2019.

F. Santambrogio. *Optimal Transport for Applied Mathematicians: Calculus of Variations, PDEs, and Modeling*. Springer, 1st edition, 2015.

Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st international conference on distributed computing systems workshops*, pp. 166–171. IEEE, 2011.

Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, Brian Cleary, Vidya Subramanian, Aryeh Solomon, Joshua Gould, Siyan Liu, Stacie Lin, Peter Berube, Lia Lee, Jenny Chen, Justin Brumbaugh, Philippe Rigollet, Konrad Hochedlinger, Rudolf Jaenisch, Aviv Regev, and Eric S. Lander. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943.e22, 2019. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2019.01.006. URL https://www.sciencedirect.com/science/article/pii/S009286741930039X.

Morgan A. Schmitz, Matthieu Heitz, Nicolas Bonneel, Fred Maurice Ngolè Mboula, David Coeurjolly, Marco Cuturi, Gabriel Peyré, and Jean-Luc Starck. Wasserstein dictionary learning: Optimal transport-based unsupervised nonlinear dictionary learning. *SIAM J. Imaging Sci.*, 11(1):643–678, 2018. doi: 10.1137/17M1140431. URL https://doi.org/10.1137/17M1140431.

Gang Sun, Alexander Hawken, and Peter Harrowell. The displacement field associated with the freezing of a melt and its role in determining crystal growth kinetics. *Proceedings of the National Academy of Sciences*, 117(7):3421–3426, 2020. ISSN 0027-8424. doi: 10.1073/pnas.1915806117. URL https://www.pnas.org/content/117/7/3421.

C. Villani. *Topics in Optimal Transportation*. American Mathematical Society, 2003.

Karren D Yang and Caroline Uhler. Scalable unbalanced optimal transport using generative adversarial networks. *arXiv preprint arXiv:1810.11447*, 2018.

6 APPENDIX

6.1 DETAILS FOR SECTION 2

Lemma 6.1. *Given a transport plan σ and a set of dual weights that are feasible with respect to σ, if any edge $(a, b) \in A \times B$ satisfies equation 2 but not equation 1, then $\sigma(a, b) = \min\{\mu_a, \nu_b\}$.*

Proof. For any feasible transport plan σ, When $\sigma(a, b) = \min\{\mu_a, \nu_b\}$, the edge (a, b) has to satisfy equation 2. In all other cases, the edge has to satisfy equation 1. Therefore, in a feasible transport plan σ, if an edge (a, b) satisfies equation 2 and not equation 1, then we can conclude $\sigma(a, b) = \min\{\mu_a, \nu_b\}$. □

Proof of Lemma 2.1: For the sake of contradiction, let σ' be another partial transport plan carrying a mass of α such that $w(\sigma') < w(\sigma)$.

First, without loss of generality, we transform σ and σ' so that σ remains a feasible transport plan, σ and σ' transport the same mass, and, $w(\sigma) - w(\sigma')$ remains unchanged. Furthermore, this transformation guarantees that the dual weights for the feasible transport plan σ are such that every edge (a, b) which carries a positive flow in σ', i.e., $\sigma'(a, b) > 0$, also satisfies equation 1. We present this transformation next.

Consider any edge (a, b) that satisfies $y(a) + y(b) > c(a, b)$, i.e., it does not satisfy equation 1 but satisfies equation 2. From Lemma 6.1 its flow $\sigma(a, b)$ is $\min\{\mu_a, \nu_b\}$. For every such edge, we reduce their flow $\sigma(a, b)$ and $\sigma'(a, b)$ by $\sigma'(a, b)$ and set

- $\sigma'(a, b) \leftarrow 0$, and,
- $\sigma(a, b) \leftarrow \{\mu_a, \nu_b\} - \sigma'(a, b)$.

We also reduce the demand at a to $\mu_a \leftarrow \mu_a - \sigma'(a, b)$ and the supply at b to $\nu_b \leftarrow \nu_b - \sigma'(a, b)$.

The transformed σ and σ' continue to transport the same mass. Moreover, their difference in costs of σ and σ' does not change due to this transformation and we are guaranteed that if the edge (a, b) has a positive flow with respect to σ', i.e., $\sigma'(a, b) > 0$ then (a, b) will satisfy equation 1. Also, note that the transformation does not change the surplus (resp. deficit) with respect to σ at any node $b \in B$ (resp. $a \in A$). We present the rest of the proof assuming that σ and σ' are transformed.

For any vertex $a \in A$ (resp. $b \in B$), let x_a (resp. x_b) denote the deficit (resp. surplus) at a (resp. b) with respect to σ. Recollect that $\mu_a = \sum_{b \in B} \sigma(a, b) + x_a$ and $\nu_b = \sum_{a \in A} \sigma(a, b) + x_b$. Using this, we can write

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B} \nu_b y(b) = \sum_{(a,b) \in A \times B} \sigma(a, b)(y(a) + y(b)) + \sum_{a \in A} y(a) x_a + \sum_{b \in B} y(b) x_b.$$

Since the transport plan σ satisfies (C), we know that if $x_a > 0$, then $y(a) = 0$ and if $x_b > 0$, then $y(b) = y_{\max}$. Using this and the fact that if $\sigma(a, b) > 0$, then $y(a) + y(b) \geq c(a, b)$ (from equation 2), we get

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B} \nu_b y(b) \geq \sum_{(a,b) \in A \times B} \sigma(a, b) c(a, b) + y_{\max} \sum_{b \in B} x_b,$$

which can be rewritten as

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B} \nu_b y(b) \geq w(\sigma) + y_{\max} \sum_{b \in B} x_b. \tag{7}$$

For any $a \in A$ (resp. $b \in B$), let x'_a (resp. x'_b) be the deficit (resp. excess) with respect to σ'. Recollect that $\mu_a = \sum_{b \in B} \sigma'(a, b) + x'_a$ and $\nu_b = \sum_{a \in A} \sigma'(a, b) + x'_b$. We can write

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B} \nu_b y(b) = \sum_{(a,b) \in A \times B} \sigma'(a, b)(y(a) + y(b)) + \sum_{a \in A} y(a) x'_a + \sum_{b \in B} y(b) x'_b. \tag{8}$$

From our initial transformation, any edge (a, b) with $\sigma'(a, b) > 0$ also satisfies equation 1, i.e., $y(a) + y(b) \leq c(a, b)$. Thus,

$$\sum_{(a,b) \in A \times B} \sigma'(a, b)(y(a) + y(b)) \leq \sum_{(a,b) \in A \times B} \sigma'(a, b) c(a, b). \tag{9}$$

Furthermore, from (C) every vertex $a \in A$ has $y(a) \leq 0$ and every vertex $b \in B$ has $y(b) \leq y_{\max}$. Using this and equation 9 in equation 8, we get

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B} \nu_b y(b) \leq \sum_{(a,b) \in A \times B} \sigma'(a,b) c(a,b) + y_{\max} \sum_{b \in B} x'_b,$$

or

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B} \nu_b y(b) \leq w(\sigma') + y_{\max} \sum_{b \in B} x'_b. \tag{10}$$

Since σ and σ' transport the same mass, the total excess across all the supply nodes in both σ and σ' will be the same, i.e., $\sum_{b \in B} x_b = \sum_{b \in B} x'_b$. Combining this with equation 7 and equation 10, we conclude that $w(\sigma') \geq w(\sigma)$ leading to a contradiction. Therefore, σ is indeed an α-optimal partial transport.

Proof of Lemma 2.2: After pushing a mass of $k \in [0, r_P]$ along P, the flow on every edge of P is updated. The flow on every other edge remains unchanged. Therefore, to show that the flow remains valid, we show

(i) The mass transported on every edge of P after augmentation remains non-negative and bounded by $\min\{\mu_a, \nu_b\}$,

(ii) The mass transported out of b (resp. into a) increases by k and does not exceed ν_b (resp. μ_a), and,

(iii) For every other node $b' \in B \setminus \{b\}$ that appears on the path P, the total mass transported out of b' remains unchanged.

Proof of (i): For any backward edge $\overrightarrow{a'b'}$ on P, its residual capacity before augmentation is equal to $\sigma(a', b')$. By construction k is at most the residual capacity of any edge on P. Therefore, after augmentation, the new flow is $\sigma(a', b') - k > 0$.

For any forward edge $\overrightarrow{a'b'}$ on P, its residual capacity before augmentation is equal to $\min\{\mu_a, \nu_b\} - \sigma(a', b')$. By construction k is at most the residual capacity of any edge on P. Therefore, after augmentation, the new flow is $\sigma(a', b') + k \leq \sigma(a', b') + (\min\{\mu_a, \nu_b\} - \sigma(a', b')) \leq \min\{\mu_a, \nu_b\}$.

Proof of (ii):

Next, we argue that the total incoming supplies at a does not exceed μ_a. By our choice, r_P is at most the deficit at a, i.e., $r_P \leq \mu_a - \sum_{b' \in B} \sigma(a, b')$. Using the fact that $k \leq r_P$, we can rewrite this as

$$\sum_{b' \in B} \sigma(a, b') + k \leq \mu_a. \tag{11}$$

After augmentation, the supplies transported to a along the last edge of the augmenting path increases by k and becomes $\sum_{b' \in B} \sigma(a, b') + k$ which from equation 11 is at most μ_a.

Next, we argue that the total outgoing supplies from b does not exceed ν_a. By our choice, r_P is at most the surplus at b, i.e., $r_P \leq \mu_b - \sum_{a' \in B} \sigma(a', b)$. Using the fact that $k \leq r_P$, we can rewrite this as

$$\sum_{a' \in A} \sigma(a', b) + k \leq \nu_b. \tag{12}$$

After augmentation, the supplies transported from b along the first edge of the augmenting path increases by k and becomes $\sum_{a' \in A} \sigma(a', b) + k$ which from equation 12 is at most ν_b.

Proof of (iii): For any other demand (resp. supply) node $a' \in A \cap P$ (resp. $b' \in B \cap P$) with $a' \neq a$ (resp. $b' \neq b$), the total supplies transported to a' (resp. from b') after the transport plan is updated remains unchanged. This is because a' (resp. b') has exactly one forward and one backward edge of P incident on it. The increase in supply transported to a' (resp. from b') via the forward edge is canceled out by the decrease in supply transported along the backward edge.

This completes the proof of the fact that after augmentation, the flow remains valid.

Finally, the change is cost of the transport plan due to augmentation is $w(\sigma_{\alpha+k}) - w(\sigma_\alpha)$. Since $\sigma_{\alpha+k}$ and σ_α only differ in the flow transported along the edges of P, we can write this difference as

$$\sum_{\overrightarrow{uv} \in P \text{ is a forward edge}} kc(u,v) - \sum_{\overrightarrow{uv} \in P \text{ is a backward edge}} kc(u,v) = k\Phi(P),$$

or,

$$w(\sigma_{\alpha+k}) = w(\sigma_\alpha) + k\Phi(P).$$

Proof of Lemma 2.3: By the definition of net-cost, we have

$$\begin{aligned}
\Phi(P) &= \sum_{\overrightarrow{uv} \in P \text{ is a forward edge}} c(u,v) - \sum_{\overrightarrow{uv} \in P \text{ is a backward edge}} c(u,v) \\
&= \sum_{\overrightarrow{uv} \in P \text{ is a forward edge}} \Big(y(u) + y(v) + s(u,v)\Big) - \sum_{\overrightarrow{uv} \in P \text{ is a backward edge}} \Big(y(u) + y(v) - s(u,v)\Big) \\
&= y(b) - y(a) + \sum_{\overrightarrow{uv} \in P} s(u,v). \qquad (13)
\end{aligned}$$

The last equality follows from the definition of slacks and from the fact that every vertex (except for a and b) appear on exactly one forward and one backward edge.

Proof of Lemma 2.4: From (C), $y(b) = y_{\max}$ and $y(a) = 0$. Since every edge $\overrightarrow{uv}$ on P is admissible, i.e., $s(u,v) = 0$, rewriting equation 5 we get $\Phi(P) = y_{\max}$.

6.2 Proofs of Section 3

In this section, we present the missing proofs from Section 3

6.2.1 Proof of Lemma 3.1:

The proof is structured as follows. We begin by showing that (i) the algorithm maintains (C) as an invariant. After completing this proof, we will show that (ii) the algorithm terminates in a finite number of steps.

Proof of Part (i): Recollect that, at the start of the first phase of our algorithm, our initial flow and dual weight assignment are feasible and satisfy (C). Inductively assume that (C) is satisfied for the first $i-1$ phases. We would like to show that (C) continues to hold at the end of phase i.

Recollect that phase i consists of two steps. In Lemma 6.2, we show that the updated dual weights satisfy equation 1 and equation 2 after Step 1. Step 2 of the algorithm finds and augments along an augmenting path consisting only of admissible edges. Since Step 2 does not modify the dual weights, all other conditions of (C) except feasibility of σ holds trivially. Therefore, we focus on showing that the new transport plan σ after augmentation remains feasible. Since the dual weights remain unchanged, every edge (u,v) in the new residual graph that was also in the old residual graph (prior to augmentation) remains feasible. To complete the argument, we show that any edge (u,v) that has been created due to augmentation satisfies the feasiblity conditions. For any newly created forward (resp. backward) edge $\overrightarrow{uv}$, prior to augmentation, the corresponding backward (resp. forward) edge $\overrightarrow{vu}$ is admissible and satisfies $y(u) + y(v) = c(u,v)$. Since dual weights remain unchanged in Step 2, the forward edge (resp. backward edge) will satisfy $y(u) + y(v) = c(u,v)$ and hence $\overrightarrow{uv}$ is feasible.

Proof of Part (ii): Next, we show that the algorithm terminates in a finite number of steps. Since all demands and supplies are rational, there is an integer L such that the demand (resp. supply) at every node $a \in A$ (resp. $b \in B$) can be expressed as d_a/L (resp. s_b/L), where d_a (resp. s_b) is an integer. Let U' be sum $\sum_{a \in A} d_a$ and $S' = \sum_{b \in B} s_b$.

We begin by showing (Lemma 6.3) that, at the end of step 1, there is at least one augmenting path consisting only of admissible (i.e., zero slack) edges. From Lemma 6.4, the bottleneck capacity of this path is r/L where r is a positive integer ≥ 1. As a result of augmentation, we will push $r/L \geq 1/L$ units of flow and reduce the total surplus and deficit by $r/L \geq 1/L$ in each phase. Thus, we will find a maximum transport plan in at most $\min\{U', S'\}$ phases, i.e., the algorithm terminates in finite number of phases.

Lemma 6.2. *The dual updates (U1) and (U2) conducted in the first step of phase i guarantee that, at the end of the first step of phase i, (C) holds.*

Proof. To show that (C) holds, we have to show that at the end of the first step,

(a) The transport plan is feasible,

(b) The dual weights for every vertex $a \in A$ (resp. $b \in B$) satisfies $y(a) \leq 0$ (resp. $y(b) \geq 0$), and,

(c) if $a \in A$ (resp. $b \in B$) is free, then $y(a) = 0$ (resp. $y(b) = y_{\max}$).

Proof of (a): We begin by showing the transport plan is feasible, i.e., the dual updates do not violate feasibility of any edge. Let $y(\cdot)$ denote the dual weights prior to update and $\tilde{y}(\cdot)$ denote the dual weights after the dual updates of the first step. Recollect ℓ_v is the shortest path distances computed by the execution of the Dijkstra's shortest path algorithm on the augmented residual graph in Step 1 of phase i.

For any directed forward edge $\overrightarrow{ba}$ in the augmented residual graph, from the shortest path property, we can express

$$\ell_b + s(b, a) \geq \ell_a. \tag{14}$$

We begin by proving that the updated dual weights for any forward edge int he residual graph $\overrightarrow{ba}$ satisfy equation 1. There are four possibilities: (i) $\ell_b < \ell_t$ and $\ell_a < \ell_t$, (ii) $\ell_b \geq \ell_t$ and $\ell_a < \ell_t$, (iii) $\ell_b < \ell_t$ and $\ell_a \geq \ell_t$ or (iv) $\ell_b \geq \ell_t$ and $\ell_a \geq \ell_t$.

For case (i), The updated dual weights $\tilde{y}(b) = y(b) + \ell_t - \ell_b$ and $\tilde{y}(a) = y(a) - \ell_t + \ell_a$ and the updated feasibility condition is $\tilde{y}(b) + \tilde{y}(a) = y(b) + y(a) + \ell_a - \ell_b$. From equation 14, $\tilde{y}(b) + \tilde{y}(a) \leq y(a) + y(b) + s(a, b) = c(a, b)$. The last equality follows from the definition of slack for a forward edge.

For case (ii), the updated dual weights are $\tilde{y}(b) = y(b)$ and $\tilde{y}(a) = y(a) - \ell_t + \ell_a$. Since the dual weight of a reduces and that of b remains unchanged, their sum only reduces and $\tilde{y}(b) + \tilde{y}(a) = y(b) + y(a) + \ell_a - \ell_t \leq c(a, b)$.

For case (iii), the updated dual weights are $\tilde{y}(b) = y(b) + \ell_t - \ell_b$ and $\tilde{y}(a) = y(a)$. Note that from equation 14 and (iii), $\ell_t - \ell_b \leq \ell_a - \ell_b \leq s(a, b)$, we have $\tilde{y}(b) + \tilde{y}(a) = y(b) + \ell_t - \ell_b + y(a) \leq y(a) + y(b) + s(a, b) = c(a, b)$. The last equality follows from the definition of slack for forward edge.

In case (iv), the dual weights of u and v are not updated by the first step of the algorithm. The edge was feasible prior to the dual update and the first step does not modify $y(u)$ or $y(v)$ and therefore, the edge continues to be feasible.

This completes the proof of feasibility for a forward edge. The arguments are identical for backward edges. For the sake of completion, we describe the proof for backward edges in case (i).

For any backward edge $\overrightarrow{ab}$ in the residual graph, and from the shortest path property on the augmented residual graph,

$$\ell_a + s(a, b) \geq \ell_b. \tag{15}$$

In case (i), the updated dual weights $\tilde{y}(a) = y(a) - \ell_t + \ell_a$ and $\tilde{y}(b) = y(b) - \ell_b + \ell_t$. As a result,

$$\begin{aligned} \tilde{y}(a) + \tilde{y}(b) &= y(a) - \ell_t + \ell_a + y(b) - \ell_b + \ell_t \\ &= y(a) + y(b) - (\ell_b - \ell_a) \\ &\geq y(a) + y(b) - s(a, b) = c(a, b). \end{aligned}$$

Therefore, every backward edge remains feasible.

Proof of (b): For any point $b \in B$ (resp. $a \in A$) that undergoes update in (U1) (resp. (U2)), since $\ell_b < \ell_t$ (resp. $\ell_a < \ell_t$), the dual weight of b (resp. a) will change by $\ell_t - \ell_b$ (resp. $\ell_a - \ell_t$) which is non-negative (resp. non-positive). Since all dual weights start at 0, the dual weights of B (resp. A) remains non-negative (resp. non-positive).

Proof of (c): In each phase, for every free vertex $b \in B$, since there is a direct edge of weight 0 from s to b in the augmented residual graph, the shortest distance from s to b is 0, i.e., $\ell_b = 0$. Therefore, b will incur the largest possible increase of ℓ_t to its dual weight (In update (U1)). Thus at any point in the algorithm, for any free vertex $b \in B$, $y(b) = \max_{v \in A \cup B} |y(v)|$. For any free point $a \in A$, since there is an edge from a to t of weight 0, $\ell_t \leq \ell_a$. Hence, no free vertex will experience a change in dual weight (given by (U2)) at any point in the algorithm and $y(a)$ remains 0. □

Lemma 6.3. *At the end of step 1, there is at least one augmenting path of admissible edges.*

Proof. Let P' be the shortest path from s to t in the augmented residual graph, and let P be the path obtained by removing s and t from this path. It is easy to see that P is an augmenting path. It suffices if we show that every forward (resp. backward) edge $\overrightarrow{ba} \in P$ (resp. $\overrightarrow{ab} \in P$) is admissible. From the optimal substructure of shortest paths, for any edge $\overrightarrow{ba} \in P$ (resp. $\overrightarrow{ab} \in P$), $\ell_a = \ell_b + s(b, a)$ (resp. $\ell_b = \ell_a + s(a, b)$).

The updated dual weight $\tilde{y}(a) = y(a) - \ell_t + \ell_a$ and $\tilde{y}(b) = y(b) + \ell_t - \ell_b$. Thus, $\tilde{y}(a) + \tilde{y}(b) = y(a) + y(b) + \ell_a - \ell_b$. Since, $\ell_a = \ell_b + s(b, a)$ (resp. $\ell_b = \ell_a + s(a, b)$), we have $\tilde{y}(a) + \tilde{y}(b) = y(a) + y(b) + s(a, b) = c(a, b)$ (resp. $\tilde{y}(a) + \tilde{y}(b) = y(a) + y(b) - s(a, b) = c(a, b)$), i.e., the slack with respect to the updated dual weights is 0. Therefore, all edges on P are admissible. □

Lemma 6.4. *At any point in the algorithm, the residual capacity of all edges as well as the surplus (resp. deficit) at any supply (resp demand) node can be expressed as r/L where r is an integer. As a result, for any phase, the augmenting path computed during the phase has a bottleneck capacity that can be expressed as r/L for some integer r.*

Proof. At the start of the algorithm, there is no backward edge in the residual network. For every forward edge $(b, a) \in B \times A$, its residual capacity is simply $\min\{s_b, d_a\}$ which is of the form r/L. Therefore, the lemma is true at the start of the algorithm.

Inductively assume that the lemma is true for until the end of phase $i - 1$.

In phase i, the only changes to the residual capacity happens due to augmentation along some augmenting path P from a free vertex b to a free vertex a. Note that the changes are addition or subtraction with the bottleneck capacity. From inductive hypothesis, for some integers r', r'', r''' the bottleneck capacity is simply the minimum of the surplus at b (which can be expressed as r'/L), the deficit at a (which can be expressed as r''/L) and the residual capacity of the bottleneck edge (which can be expressed as r'''/L). Thus, the bottleneck capacity of P is $\overline{r}/L$ where $\overline{r} = \min\{r', r'', r'''\}$ is an integer. Augmentation will increase or reduce the residual capacities (as well as the surplus and deficit) by $\overline{r}/L$. The residual capacities after addition or subtraction by $\overline{r}/L$ will continue to satisfy the conditions of the Lemma. □

6.2.2 Proof of Lemma 3.2

Proof of Lemma 3.2(a): We give a proof by induction. *Base case:* When $i = 1$, we would have transported a mass of r_1 along P_1. From Lemma 2.2, the cost of σ_1 is

$$w(\sigma_1) = w(\sigma_0) + r_1 \Phi(P_1). \tag{16}$$

From Lemma 2.4 and (C), we have $\Phi(P_1) = y^1_{\max}$. Therefore, we can rewrite equation 16 as $w(\sigma_1) = w(\sigma_0) + r_1 y^1_{\max} = r_1 y^1_{\max}$. The last inequality follows from the fact that σ_0 is an empty transport plan.

Inductive Step: Inductively assume that $w(\sigma_{i-1}) = \sum_{j=1}^{i-1} r_j y^j_{\max}$. From Lemma equation 2.4 and (C), we have $y^i_{\max} = \Phi(P_i)$. Since σ_i is the transport plan obtained after augmenting σ_{i-1} along P_i with a mass of r_i, the cost of σ_i is $w(\sigma_i) = w(\sigma_{i-1}) + r_i \Phi(P_i) = w(\sigma_{i-1}) + r_i y^i_{\max} =$

$\sum_{j=1}^{i-1} r_j y^j_{\max} + r_i y^i_{\max}$. The last inequality follows from our inductive hypotheses. Therefore, we can write $w(\sigma_i)$ as $\sum_{j=1}^{i} r_j y^j_{\max}$ concluding our proof.

Proof of Lemma 3.2(b): At the start of phase i, from (C), the dual weight of any free supply vertex $b \in B$ is $y^{i-1}_{\max}$, i.e., $y(b) = y^{i-1}_{\max}$. By its construction, there is an edge $\overrightarrow{sb}$ with weight 0 in the augmented residual graph. Therefore the shortest path length ℓ_b from s to b as computed in the first step of the algorithm is 0. If $\ell_t = \ell_b = 0$, then the dual weight $y(b)$ remains unchanged. Since (C) holds during the execution of the algorithm, $y^i_{\max} = y(b) = y^{i-1}_{\max}$. Otherwise, if $\ell_b < \ell_t$, the dual update (U1) will increase the dual weight of b. Since (C) holds during the execution of the algorithm, $y^i_{\max} = y(b) + \ell_t - \ell_b = y^{i-1}_{\max} + \ell_t$.

6.2.3 Proof of Lemma 3.3

For any intermediate value $\alpha \in (\alpha_{i-1}, \alpha_i)$, we can construct the α-optimal partial transport plan by pushing $(\alpha - \alpha_{i-1})$ mass along P_i. The resulting transport plan will transport will transport a mass of α and also satisfy (C). Therefore, from Lemma 2.1 and Lemma 3.1, it is an α-optimal partial transport plan. From Lemma 2.2, its cost will be $w(\sigma_{i-1}) + (\alpha - \alpha_{i-1})\Phi(P_i) = \sum_{j=1}^{i-1} r_j y^j_{\max} + (\alpha - \alpha_{i-1}) y^i_{\max}$.

6.2.4 Proof of Lemma 3.5

If all optimal partial transports are unique, then from Lemma 6.5, there is no augmenting path of admissible edges after any phase i. Since there is no augmenting path in the admissible graph, from Lemma 6.6, we conclude that the slope of (p_i, p_{i+1}) is strictly greater than (p_{i-1}, p_i), i.e., the OT-profile generated by our algorithm does not have three consecutive collinear points. This implies that our algorithm executes exactly K phases, where K is the complexity of the OT-profile and the running time of our algorithm is $O(n^2 K)$.

Lemma 6.5. *For any input μ and ν, suppose all optimal partial transports are unique. Then after each phase of our exact algorithm, there is no augmenting path of admissible edges.*

Proof. For the sake of contradiction, consider that at the end of phase i, there is still an augmenting path of admissible edges. Let σ be the transport plan at the start of phase i and let σ transport a mass of α. Let P be the augmenting path computed in phase i and let P' be an augmenting path that remains in the admissible graph at the end of phase i. Let r be the bottleneck capacity of P and let r' be the bottleneck capacity of P'. We will construct two different partial transports σ_1 and σ_2 both of which are optimal transport plans and that transport a mass of $\alpha + \min\{r, r'\}$. This contradicts the uniqueness assumption.

Constructing σ_1: To construct σ_1, we simply transport $\min\{r, r'\}$ along the path P. Note that the resulting transport plan along with the dual weights satisfies (C). Thus, from Lemma 2.1, σ_1 is an $\alpha + \min\{r, r'\}$-optimal partial transport plan.

Constructing σ_2: To construct σ_2, we simply transport $\min\{r, r'\}/2$ along P. After this, we will show that P' will be an augmenting path with respect to this new transport plan and has a bottleneck capacity of at least $\min\{r, r'\}/2$. We then push a mass of $\min\{r, r'\}/2$ along P'. The resulting plan σ_2 also transports a mass of $\alpha + \min\{r, r'\}$. Similar to σ_1, the resulting transport plan continues to satisfy (C) and from Lemma 2.1, σ_2 is an $\alpha + \min\{r, r'\}$-optimal partial transport.

To complete our argument, we need to show that after pushing $\min\{r, r'\}/2$ units of flow along P, P' is an augmenting path with a bottleneck capacity of at least $\min\{r, r'\}/2$. To show this, it suffices if we show that every edge of P' is indeed in the admissible graph and has a residual capacity of at least $\min\{r, r'\}/2$. There are three possibilities. For any edge $\overrightarrow{uv}$ in P', (i) $\overrightarrow{uv}$ is not in P, or, (ii) $\overrightarrow{uv}$ is in P, or, (iii) $\overrightarrow{vu} \in P$.

For case (i), the residual capacity of $\overrightarrow{uv}$ is greater than the bottleneck residual capacity of P' which is at least $r' \geq \min\{r, r'\}/2$.

In case(ii), $\overrightarrow{uv}$ is also an edge in P. Prior to augmentation, $\overrightarrow{uv}$ had a residual capacity of at least r. Therefore, after pushing a mass of $\min\{r, r'\}/2$, $\overrightarrow{uv}$ continues to have a residual capacity of at least $r - \min\{r, r'\}/2 \geq \min\{r, r'\}/2$.

In case (iii), $\overrightarrow{vu}$ is an edge on P. The residual capacity of $\overrightarrow{uv}$ is at least equal to the mass pushed on the edge $\overrightarrow{vu}$ along P, i.e., $\min\{r, r'\}/2$. Therefore, P' is an augmenting path with a bottleneck residual capacity of at least $\min\{r, r'\}/2$. □

Lemma 6.6. *Suppose there is no augmenting path of admissible edges at the end of phase i. Then, the slope of (p_i, p_{i+1}) is strictly greater than the slope of (p_{i-1}, p_i).*

Proof. Recollect that in the first step of the algorithm, if the ℓ_t value computed by Dijkstra's algorithm is 0, then the admissible graph remains unchanged. So, if there are no augmenting path of admissible edges at the end of phase i and there is at least augmenting path at the end of step 1 of phase $i+1$, we can conclude that the admissible graph has changed implying that the ℓ_t value computed in the first step of phase $i+1$ is greater than 0.

Next, note that, the change in $y_{\max}$-value after step 1 of phase $i+1$ is given by $y^{i+1}_{\max} = y^i_{\max} + \ell_t$. This is because, ℓ_b for any free vertex $b \in B_F$ is 0 and therefore, (U1) increases the dual weight of every such free vertex by $\ell_t - \ell_b = \ell_t$. Since the dual weight of any free vertex $b \in B_F$ at the end of phase i is $y(b) = y^i_{\max}$ (from (C)), we have, $y^{i+1}_{\max} = y^i_{\max} + \ell_t$. The lemma follows from the fact that $\ell_t > 0$ and $y^{i+1}_{\max}$ and $y^i_{\max}$ are the slopes of (p_i, p_{i+1}) and (p_{i-1}, p_i) respectively. □

6.3 Proof of Lemma 1.1

To prove (A), we need to show that the $(\alpha^* - \varepsilon)$-optimal partial transport plan between μ and ν generated by our algorithm does not transport any mass from the outlier points. Let ν^+ be the mass distribution given by the points in B^+. First, consider executing our exact algorithm for the distributions μ and ν^+. Let σ be the $(\alpha^* - \varepsilon)$-optimal partial transport generated by our algorithm and let $y(\cdot)$ be the corresponding set of dual weights. Let $y_{\max} = \max_{v \in A \cup B^+} |y(v)|$.

We will now show that σ is also a $(\alpha^* - \varepsilon)$-optimal partial transport between μ and ν, implying (A). We prove this as follows:

First, we show, in Lemma 6.7, that $y_{\max}$ is at most w/ε. Next, we convert σ to be a transport plan between μ and ν by assigning the points in B^- a dual weight of $y_{\max}$. In Lemma 6.8, we show that this assignment will make σ a feasible transport plan between μ and ν that satisfies (C). By Lemma 2.1, σ will be a $(\alpha^* - \varepsilon)$-optimal partial transport plan between μ and ν.

Lemma 6.7. *Let σ be the $(\alpha^* - \varepsilon)$-optimal partial transport plan between μ and ν^+ generated by our algorithm along with the dual weights $y(\cdot)$ that satisfy (C). Then, For every $v \in B^+ \cup A$, $|y(v)| \leq w/\varepsilon$.*

Lemma 6.8. *Let σ be the $(\alpha^* - \varepsilon)$-optimal partial transport generated by our algorithm between distributions μ and ν^+ and let $y(\cdot)$ be the corresponding dual weights satisfying (C). Then, σ is also and $(\alpha^* - \varepsilon)$-optimal partial transport between μ and ν. Furthermore, $D\omega(\alpha^* - \varepsilon) = y_{\max}$.*

Next, we prove (B). Note that Lemma 6.7 and 6.8 already establish that $D\omega(\alpha^* - \varepsilon) \leq w/\varepsilon$. To show that $D\omega(\alpha^* + \varepsilon) \geq Cw/\varepsilon$, consider executing our exact algorithm for the distribution μ and ν. Consider the $(\alpha^* + \varepsilon)$-optimal partial transport σ' and dual weights $y'(\cdot)$ satisfying (C) generated by our algorithm. Let $y'_{\max} = \max_{v \in A \cup B^+} |y'(v)|$. Since the total mass of the points in B^+ is α^*, σ' will also transport some mass from B^-. Consider any edge $(a, b) \in A \times B^-$ with $\sigma'(a, b) > 0$. By our assumption $c(a, b) \geq Cw/\varepsilon$. Since $\sigma'(a, b) > 0$, by equation 2, $y'(b) + y'(a) \geq c(a, b) \geq Cw/\varepsilon$. Since, from (C), $y'(a) \leq 0$, we have $y'_{\max} \geq y'(b) \geq Cw/\varepsilon$. From Lemma 3.3, it follows that $D\omega(\alpha^* + \varepsilon) \geq Cw/\varepsilon$.

6.3.1 Proof of Lemma 6.7

Let σ^* be the optimal transport plan between μ and ν^+. Recollect that w is the cost of σ^*.

First, without loss of generality, we transform σ and σ^* so that σ remains a feasible transport plan, σ and σ^* transport the same mass, and, $w(\sigma) - w(\sigma^*)$ remains unchanged. Furthermore, the total surplus across all nodes of B^+ with respect to the transport plan σ remains at ε. Finally, this transformation guarantees that the dual weights for the feasible transport plan σ are such that every edge (a, b) which carries a positive flow in σ^*, i.e., $\sigma^*(a, b) > 0$, also satisfies equation 1. We present this transformation next.

Consider any edge (a, b) that satisfies $y(a) + y(b) > c(a, b)$, i.e., it does not satisfy equation 1 but satisfies equation 2. From Lemma 6.1 its flow $\sigma(a, b)$ is $\min\{\mu_a, \nu_b\}$. For every such edge, we reduce their flow $\sigma(a, b)$ and $\sigma^*(a, b)$ by $\sigma^*(a, b)$ and set

- $\sigma^*(a, b) \leftarrow 0$, and,
- $\sigma(a, b) \leftarrow \{\mu_a, \nu_b\} - \sigma^*(a, b)$.

We also reduce the demand at a to $\mu_a \leftarrow \mu_a - \sigma^*(a, b)$ and the supply at b to $\nu_b \leftarrow \nu_b - \sigma^*(a, b)$. Note that this transformation does not change the total surplus with respect to σ at the nodes of B^+. We present our arguments with respect to the transformed σ and σ^*.

For any free vertex $a \in A_F$ (resp. $b \in B_F$), let x_a (resp. x_b) denote the deficit (resp. surplus) at a (resp. b) with respect to the transport plan σ. Recollect that $\mu_a = \sum_{b \in B^+} \sigma(a, b) + x_a$ and $\nu_b = \sum_{a \in A} \sigma(a, b) + x_b$. Using this, we can write

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B^+} \nu_b y(b) = \sum_{(a,b) \in A \times B^+} \sigma(a, b)(y(a) + y(b)) + \sum_{a \in A} y(a) x_a + \sum_{b \in B^+} y(b) x_b.$$

Since the transport plan σ satisfies (C), we know that if $x_a > 0$, then $y(a) = 0$ and if $x_b > 0$, then $y(b) = y_{\max}$. Using this and the fact that if $\sigma(a, b) > 0$, then $y(a) + y(b) \geq c(a, b)$ (from equation 2), we get

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B^+} \nu_b y(b) \geq \sum_{(a,b) \in A \times B^+} \sigma(a, b) c(a, b) + y_{\max} \sum_{b \in B^+} x_b,$$

which can be rewritten as

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B^+} \nu_b y(b) \geq w(\sigma) + y_{\max} \sum_{b \in B^+} x_b. \tag{17}$$

Recollect that the total surplus with respect to σ is ε, i.e., $\sum_{b \in B^+} x_b = \varepsilon$. Therefore, we can rewrite equation 17 as

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B^+} \nu_b y(b) \geq w(\sigma) + y_{\max} \varepsilon. \tag{18}$$

For any $a \in A$, let x'_a be the deficit with respect to σ^*. Since σ^* transports all supplies, the surplus at any node $b \in B^+$ with respect to σ^* is 0. Recollect that $\mu_a = x'_a + \sum_{b \in B^+} \sigma^*(a, b)$ and $\nu_b = \sum_{a \in A} \sigma^*(a, b)$. We can write

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B^+} \nu_b y(b) = \sum_{(a,b) \in A \times B^+} \sigma^*(a, b)(y(a) + y(b)) + \sum_{a \in A} y(a) x'_a. \tag{19}$$

From our initial transformation, any edge (a, b) with $\sigma^*(a, b) > 0$ also satisfies equation 1, i.e., $y(a) + y(b) \leq c(a, b)$. Thus,

$$\sum_{(a,b) \in A \times B^+} \sigma^*(a, b)(y(a) + y(b)) \leq \sum_{(a,b) \in A \times B^+} \sigma^*(a, b) c(a, b). \tag{20}$$

Furthermore, from (C) every vertex $a \in A$ has $y(a) \leq 0$. Using this and equation 20 in equation 19, we get

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B^+} \nu_b y(b) \leq \sum_{(a,b) \in A \times B^+} \sigma^*(a, b) c(a, b),$$

or

$$\sum_{a \in A} \mu_a y(a) + \sum_{b \in B^+} \nu_b y(b) \leq w(\sigma^*) \leq w. \tag{21}$$

Combining equation 18 with equation 21, we get $w(\sigma) + \varepsilon y_{\max} \leq w$. Using the fact that $w(\sigma) \geq 0$, we get

$$y_{\max} \leq w/\varepsilon.$$

For every $v \in A \cup B^+$, from the definition of $y_{\max}$, we have $|y(v)| \leq y_{\max}$ $|y(v)| \leq y_{\max} \leq w/\varepsilon$ as desired.

6.3.2 PROOF OF LEMMA 6.8

Let i be the phase where σ is generated by our algorithm. Let σ' be the transport plan before the execution of Step 2 of phase i. Since Step 2 does not update the dual weights, $y(\cdot)$ will also be the dual weights after Step 1 of phase i and σ' and $y(\cdot)$ will form a feasible transport plan. Let $\beta < \alpha^* - \varepsilon$ be the mass transported by σ'.

For every vertex $b \in B^-$, suppose we assign it a dual weight $y(b) = y_{\max}$. By Lemma 6.7,

$$y(b) \leq w/\varepsilon. \tag{22}$$

We show that every forward edge $\overrightarrow{ba}$ remains feasible.[5] By the outlier assumption, every edge $(a, b) \in A \times B^-$, $c(a, b) > w/\varepsilon$. From (C), the dual weight $y(a) \leq 0$ and from equation 22 $y(b) \leq w/\varepsilon$. Therefore their sum $y(a) + y(b) \leq w/\varepsilon \leq c(a, b)$.

Thus both σ and σ' are feasible transport plans between ν and μ and satisfy (C), i.e., both σ and σ' are optimal partial transports. So, the points $(\beta, w(\sigma'))$ and $(\alpha^* - \varepsilon, w(\sigma))$ are on the OT-profile ω. Therefore, the slope at $(\alpha^* - \varepsilon)$ is precisely the slope of the line connecting these two points which is given by

$$\frac{w(\sigma) - w(\sigma')}{(\alpha^* - \varepsilon) - \beta}.$$

Since the net-cost of the augmenting path in phase i is equal to $y_{\max}$ and since step 2 pushes a mass of $((\alpha^* - \varepsilon) - \beta)$, we can write the difference $w(\sigma) - w(\sigma') = y_{\max}((\alpha^* - \varepsilon) - \beta)$. Therefore, the slope $D\omega(\alpha^* - \varepsilon) = y_{\max} \leq w/\varepsilon$ (From Lemma 6.7).

7 APPROXIMATION ALGORITHM DETAILS

In this section, we describe how the LMR-algorithm can be adapted to compute an approximate OT-profile $\overline{\omega}$. We also describe a function $D\overline{\omega}$ that satisfies an approximate outlier lemma.

7.1 SCALING DEMANDS AND SUPPLIES

In this section, we describe the scale-and-round step of the LMR algorithm that converts the demands and supplies to integers. The LMR-algorithm then finds an approximate solution to the transformed problem in $O(\frac{n^2}{\delta} + \frac{n}{\delta^2})$ time. Finally, this solution is mapped back to the original demands and supplies. The total loss in accuracy in the cost due to this transformation (steps 1 and 3) is at most $S\delta/2$.

Set $\theta = \frac{4n}{S\delta}$. Let $\mathcal{I}$ be the input instance for the optimal transport problem with each demand location $a \in A$ having a demand of μ_a and each supply location $b \in B$ having a supply of ν_b. We create a new input instance $\mathcal{I}'$ by scaling the demand at each node $a \in A$ to $\overline{d}_a = \lceil \mu_a \theta \rceil$ and scaling the supply at each node $b \in B$ to $\overline{s}_b = \lfloor \nu_b \theta \rfloor$. Let the total supply be $\mathcal{S} = \sum_{b \in B} \overline{s}_b$. Since we scale the supplies by θ and round them down, we have

$$\mathcal{S} = \sum_{b \in B} \overline{s}_b = \sum_{b \in B} \lfloor \nu_b \theta \rfloor \leq \theta \sum_{b \in B} \nu_b = \theta S. \tag{23}$$

We also have

$$\mathcal{S} + n = n + \sum_{b \in B} \overline{s}_b \geq \sum_{b \in B} \lceil \nu_b \theta \rceil \geq \theta \sum_{b \in B} \nu_b = \theta S. \tag{24}$$

Recollect that for any input $\mathcal{I}$ to the optimal transport problem, the total supply S is no more than the total demand U. Since the new supplies are scaled by θ and rounded down whereas the new demands are scaled by θ and rounded up, the total supplies in $\mathcal{I}'$ remains no more than the total demand. For any $\alpha \in [0, S]$, let σ' be any feasible $\min\{\mathcal{S}, \alpha\theta\}$-partial transport plan for $\mathcal{I}'$. Now consider a transport plan σ that sets, for each edge (a, b), $\sigma(a, b) = \sigma'(a, b)/\theta$. As described below, the transport plan σ is not necessarily feasible or necessarily an α-partial transport plan for $\mathcal{I}$.

[5] There is no backward edge incident on b since both σ and σ' only transport mass on edges of $A \times B^+$.

(i) When $\mathcal{S}/\theta < \alpha$, the transport plan σ will transport a mass of $\mathcal{S}/\theta < \alpha$. Since $\alpha \in [0, S]$, $\alpha - \mathcal{S}/\theta \leq S - \mathcal{S}/\theta \leq n/\theta$. The last inequality follows from equation 24.

(ii) σ is not a feasible plan for $\mathcal{I}$ since the total demand met at any node $a \in A$ can be more than μ_a, i.e., $\sum_{b \in B} \sigma(a, b) = \sum_{b \in B} \sigma'(a, b)/\theta = \overline{d}_a/\theta = \lceil \theta \mu_a \rceil / \theta \geq \mu_a$. Note that the excess supply that reaches node $a \in A$, $\kappa_a \leq \lceil \theta \mu_a \rceil / \theta - \mu_a \leq \frac{\theta \mu_a + 1}{\theta} - \mu_a = 1/\theta$.

The cost of σ, $w(\sigma) = w(\sigma')/\theta$. We can convert σ to a feasible and α-partial transport plan for $\mathcal{I}$ in two steps.

First, one can convert σ to a feasible solution. The excess supply κ_a that reaches a demand node $a \in A$ can be removed by iteratively picking an arbitrary edge incident on a, say the edge (a, b), and reducing $\sigma(a, b)$ as well as κ_a by $\min\{\kappa_a, \sigma(a, b)\}$. This iterative process is applied until κ_a reduces to 0. This step is also repeated at every demand node $a \in A$ with a $\kappa_a > 0$. The total excess supply pushed back will increase the leftover supply at the supply nodes by $\sum_{a \in A} \kappa_a \leq n/\theta$. Combined with the left-over supply from (i), the total supply transported by σ is at least $\alpha - 2n/\theta$. σ is now a feasible transportation plan. Since the supplies transported along edges only reduce, the cost $w(\sigma) \leq w(\sigma')/\theta$.

Second, to convert this feasible plan σ to an α-partial transport plan, one can simply match the remaining at most $(\sum_{(a,b) \in A \times B} \sigma(a, b)) - \alpha$ supplies arbitrarily to leftover demands at a cost of at most 1 per unit of supply. Since $(\sum_{(a,b) \in A \times B} \sigma(a, b)) - \alpha$ is at most $2n/\theta$, the cost of this new transport plan increases by at most $2n/\theta$. Plugging in the value of θ, we get

$$w(\sigma) \leq w(\sigma')/\theta + \frac{2n}{\theta} \leq w(\sigma')/\theta + S\delta/2. \tag{25}$$

Let σ^* is the α-optimal partial transport for $\mathcal{I}$. Let σ'_{OPT} be the $\min\{\mathcal{S}, \alpha\theta\}$-optimal partial transport for input instance $\mathcal{I}'$. In Lemma 7.1, we show that $w(\sigma'_{\mathrm{OPT}}) \leq \alpha w(\sigma^*)$. In the Section 7.2, we describe how the LMR algorithm computes, for every $\alpha \in [0, S]$, σ' that transports a mass of $\min\{\mathcal{S}, \alpha\theta\}$ with a cost of $w(\sigma') \leq w(\sigma'_{\mathrm{OPT}}) + \mathcal{S}\delta/2$. From Lemma 7.1(proof of which is given in Lahn et al. (2019)), this can be rewritten as $w(\sigma') \leq \theta w(\sigma^*) + \mathcal{S}\delta/2$. By combining this with equations equation 23 and equation 25, the solution produced by our algorithm is $w(\sigma) \leq w(\sigma^*) + \mathcal{S}\delta/2\theta + S\delta/2 \leq w(\sigma^*) + S\delta/2 + S\delta/2 = w(\sigma^*) + S\delta$.

Lemma 7.1. *Let $\alpha \in [0, S]$, be a parameter. Let $\mathcal{I}$ be the original instance of the transportation problem and let $\mathcal{I}'$ be an instance scaled by θ. Let σ^* be the α-optimal partial transport plan for $\mathcal{I}$ and let σ'_{OPT} be a $\min\{\theta\alpha, \mathcal{S}\}$-optimal partial transport plan for $\mathcal{I}'$. Then $w(\sigma'_{\mathrm{OPT}}) \leq \theta w(\sigma^*)$.*

The algorithm in Section 7.2 returns an approximate OT-profile ω' for the instance $\mathcal{I}'$. For every $\alpha \in [0, S]$, $\omega'(\alpha) = w(\sigma')$; where σ' is a transport plan that transports a mass of $\min\{\mathcal{S}, \alpha\theta\}$ and has a cost of $w(\sigma') \leq w(\sigma'_{\mathrm{OPT}}) + \mathcal{S}\delta/2$. Based on the discussion above, we can convert it into a δ-approximate OT profile $\overline{\omega}$ for $\mathcal{I}$ by simply setting $\overline{\omega}(\alpha) = \omega'(\alpha)/\theta + S\delta/2$.

7.2 Algorithm for scaled demands and supplies

The input $\mathcal{I}'$ consists of a set of demand nodes A with demand of $\overline{d}_a$ for each node $a \in A$ and a set of supply nodes B with supply of $\overline{s}_b$ for each node $b \in B$ along with the cost matrix as input. Let σ'_{OPT} be the optimal transport plan for $\mathcal{I}'$. The LMR-algorithm can be modified to produce an approximate OT-profile in $O(n^2/\delta + n/\delta^2)$ time.

Definitions and notations: Let $\overline{c}(a, b) = \lfloor 4c(a, b)/\delta \rfloor$ be the scaled cost of any edge (a, b). Recollect that $w(\sigma)$ is the cost of any transport plan σ with respect to $c(\cdot, \cdot)$. Similarly, we use $\overline{w}(\sigma)$ to denote the cost of any transport plan with respect to the $\overline{c}(\cdot, \cdot)$.

This algorithm is based on a primal-dual approach. The algorithm, at all times, maintains a transport plan that satisfies the dual feasibility conditions. Given a transport plan σ along with a dual weight $y(v)$ for every $v \in A \cup B$, we say that $\sigma, y(\cdot)$ is 1-feasible if, for any two nodes $a \in A$ and $b \in B$,

$$y(a) + y(b) \leq \overline{c}(a, b) + 1 \qquad \text{if } \sigma(a, b) < \min\{\overline{s}_b, \overline{d}_a\} \tag{26}$$

$$y(a) + y(b) \geq \overline{c}(a, b) \qquad \text{if } \sigma(a, b) > 0. \tag{27}$$

These feasibility conditions is a relaxation of the one in Section 3 for costs that are scaled by $4/\delta$ and rounded down. Particularly, the +1 on the RHS of equation 26 is a relaxation of equation 1. We refer to a 1-feasible transport plan that is maximum as a 1-optimal transport plan. Consider any 1-feasible transport plan σ such that for every demand node $a \in A$ (resp. $b \in B$),

(C') The dual weight $y(a) \leq 0$ (resp. $y(b) \geq 0$) and, if a (resp. b) is a free demand (resp. supply) node, then $y(a) = 0$ (resp. $y(b) = y_{\max}$).

For any $\alpha \in [0, S]$, in Lemma 7.2, we show that any 1-feasible transport plan σ with dual weights $y(\cdot)$ satisfying (C') and transporting a mass of $\min\{\mathcal{S}, \alpha\theta\}$ satisfies $w(\sigma) \leq w(\sigma'_{\mathrm{OPT}}) + \delta\mathcal{S}/2$; here σ'_{OPT} is the $\min\{\mathcal{S}, \alpha\theta\}$-optimal partial transport as desired.

Lemma 7.2. *For any $\alpha \in [0, S]$ Let σ along with dual weights $y(\cdot)$ be a 1-feasible transport plan that satisfies (C') and transports a mass of $\overline{\alpha} = \min\{\mathcal{S}, \alpha\theta\}$. Let $\sigma' = \sigma'_{\mathrm{OPT}}$ be an $\overline{\alpha}$-optimal partial transport plan. Then, $w(\sigma) \leq w(\sigma') + \delta\mathcal{S}/2$.*

Proof. For the sake of contradiction, let σ' be another partial transport plan carrying a mass of α such that $w(\sigma') + \mathcal{S}\delta/2 < w(\sigma)$.

First, without loss of generality, we transform σ and σ' so that σ remains a 1-feasible transport plan and $w(\sigma) - w(\sigma')$ remains unchanged . Furthermore, this transformation guarantees that the dual weights for the 1-feasible transport plan σ is such that every edge (a, b) for which the optimal transport plan has a positive flow, i.e., $\sigma'(a, b) > 0$, also satisfies equation 26. We present this transformation next.

If the dual weights for an edge (a, b) do not satisfy equation 26, then using an argument similar to Lemma 6.1, the mass transported on $\sigma(a, b)$ is $\min\{\overline{s}_b, \overline{d}_a\}$. We reduce $\overline{d}_a$ and $\overline{s}_b$ by $\sigma'(a, b)$ and also reduce the mass transported on the edge (a, b) in σ' to be 0, i.e., $\sigma'(a, b) = 0$ and the mass transported on $\sigma(a, b)$ to $\min\{\overline{s}_b, \overline{d}_a\} - \sigma'(a, b)$. The transformed σ continues to be 1-feasible transport plan for the new demands and supplies. Moreover, the difference in costs of σ and σ' does not change due to this transformation and we are guaranteed that if the edge (a, b) has a positive mass with respect to σ', i.e., $\sigma'(a, b) > 0$ then (a, b) will satisfy equation 26. We present the rest of the proof assuming that σ and σ' are transformed.

For any point $a \in A$ (resp. $b \in B$), let x_a (resp. x_b) denote the deficit at a (resp. b) with respect to σ. Recollect that $\mu_a = \sum_{b \in B} \sigma(a, b) + x_a$ and $\nu_b = \sum_{a \in A} \sigma(a, b) + x_b$. Using this, we can write

$$\sum_{a \in A} \overline{d}_a y(a) + \sum_{b \in B} \overline{s}_b y(b) = \sum_{(a,b) \in A \times B} \sigma(a, b)(y(a) + y(b)) + \sum_{a \in A} y(a) x_a + \sum_{b \in B} y(b) x_b.$$

From (C'), we conclude that if $x_a > 0$, then $y(a) = 0$ and if $x_b > 0$, then $y(b) = y_{\max}$. Using this and the fact that if $\sigma(a, b) > 0$, then $y(a) + y(b) \geq \overline{c}(a, b)$, we get

$$\sum_{a \in A} \overline{d}_a y(a) + \sum_{b \in B} \overline{s}_b y(b) \geq \sum_{(a,b) \in A \times B} \sigma(a, b) \overline{c}(a, b) + y_{\max} \sum_{b \in B} x_b,$$

which can be rewritten as

$$\sum_{a \in A} \overline{d}_a y(a) + \sum_{b \in B} \overline{s}_b y(b) \geq \overline{w}(\sigma) + y_{\max}(\mathcal{S} - \overline{\alpha}). \tag{28}$$

Let x'_a and x'_b be the deficit and excess with respect to σ'. We can write

$$\sum_{a \in A} \overline{d}_a y(a) + \sum_{b \in B} \overline{s}_b y(b) = \sum_{(a,b) \in A \times B} \sigma'(a, b)(y(a) + y(b)) + \sum_{a \in A} y(a) x'_a + \sum_{b \in B} y(b) x'_b.$$

Due to the transformation, for any edge (a, b) with $\sigma'(a, b) > 0$, $y(a) + y(b) \leq \overline{c}(a, b) + 1$. Furthermore, from (C') every vertex $a \in A$ has $y(a) \leq 0$ and every vertex $b \in B$ has $y(b) \leq y_{\max}$. Using these inequalities, we get

$$\sum_{a \in A} \overline{d}_a y(a) + \sum_{b \in B} \overline{s}_b y(b) \leq (\sum_{(a,b) \in A \times B} \sigma'(a, b) \overline{c}(a, b)) + \overline{\alpha} + y_{\max} \sum_{b \in B} x_b,$$

or

$$\sum_{a \in A} \overline{d}_a y(a) + \sum_{b \in B} \overline{s}_b y(b) \leq \overline{w}(\sigma') + \overline{\alpha} + y_{\max}(\mathcal{S} - \overline{\alpha}). \tag{29}$$

From equation 28 and equation 29, we get $\overline{w}(\sigma) \leq \overline{w}(\sigma') + \overline{\alpha} \leq \overline{w}(\sigma') + \mathcal{S}$. Rescaling the input costs, we get $w(\sigma') + \mathcal{S}\delta/2 \geq w(\sigma)$. This contradicts our assumption that $w(\sigma') + \mathcal{S}\delta/2 < w(\sigma)$. □

In the rest of this section, we show that, for every value of $\alpha \in [0, S]$, the LMR algorithm incrementally constructs a 1-feasible transport plan transporting a mass of $\min\{\mathcal{S}, \alpha\theta\}$ and satisfying (C). An approximate OT-profile is generated by simply using the cost of every intermediate transport plan maintained by the LMR-algorithm. To assist in describing this algorithm, we introduce a few definitions.

For any 1-feasible transport plan σ, we can construct a directed *residual graph* identical to the one described in Section 2. We set the cost of any edge between a and b regardless of their direction to be $\overline{c}(a, b) = \lfloor 4c(a, b)/\delta \rfloor$. We incorporate the relaxation in equation 26 and redefine the slack on any edge between a and b in the residual network as

$$s(a, b) = \overline{c}(a, b) + 1 - y(a) - y(b) \qquad \text{if } (a, b) \text{ is a forward edge,} \tag{30}$$

$$s(a, b) = y(a) + y(b) - \overline{c}(a, b) \qquad \text{if } (a, b) \text{ is a backward edge} \tag{31}$$

Finally, we define any edge (a, b) in $\overline{G}_\sigma$ as admissible if $s(a, b) = 0$. The *admissible graph* $\overline{\mathcal{A}}_\sigma$ is the subgraph of $\overline{G}_\sigma$ consisting of the admissible edges of the residual graph.

7.2.1 THE ALGORITHM

Initially σ is a transport plan where, for every edge $(a, b) \in A \times B$, $\sigma(a, b) = 0$. We set the dual weights of every vertex $v \in A \cup B$ to 0, i.e., $y(v) = 0$. Note that σ and $y(\cdot)$ together form a 1-feasible transportation plan. Our algorithm executes in *phases* and terminates when σ becomes a maximum transport plan. Within each phase there are two *steps*. These steps are very similar to those in the exact algorithm. In the first step, the algorithm conducts a Hungarian Search and adjusts the dual weights so that there is at least one augmenting path of admissible edges. In the second step, the algorithm computes at least one augmenting path and updates σ by augmenting it along all paths computed. At the end of the second step, we guarantee that there is no augmenting path of admissible edges. The details are presented next.

First step (Hungarian Search): To conduct a Hungarian Search, we add two additional vertices s and t to the residual network. We add edges directed from s to every free supply node, i.e., nodes with $\sum_{a \in A} \sigma(a, b) < \overline{s}_b$. We add edges from every free demand vertex to t. All edges incident on s and t are given a weight 0. The weight of every other edge (a, b) of the residual network is set to its slack $s(a, b)$ based on its direction. We refer to the residual graph with the additional two vertices as the *augmented residual network* and denote it by $\mathcal{G}_\sigma$. We execute Dijkstra's algorithm from s in the augmented residual network $\mathcal{G}_\sigma$. For any vertex $v \in A \cup B$, let ℓ_v be the shortest path from s to v in $\mathcal{G}_\sigma$. Next, the algorithm performs a dual weight adjustment. For any vertex $v \in A \cup B$, if $\ell_v \geq \ell_t$, the dual weight of v remains unchanged. Otherwise, if $\ell_v < \ell_t$, we update the dual weight as follows: **(U1'):** If $v \in A$, we set $y(v) \leftarrow y(v) - \ell_t + \ell_v$, **(U2'):** Otherwise, if $v \in B$, we set $y(v) \leftarrow y(v) + \ell_t - \ell_v$.

This completes the description of the first step of the algorithm. The dual updates guarantee that, at the end of this step, the transport plan σ along with the updated dual weights remain 1-feasible and there is at least one augmenting path in the admissible graph.

The second step of the algorithm conducts multiple *partial DFS* step to discover one or more augmenting paths in the admissible graph (See Lahn et al. (2019) for details of this step). The transport plan is then augmented along each of these paths. At the end of this step, it is shown that there are no more augmenting paths in the residual graph.

Invariants: The following invariants were shown to hold during the execution of the algorithm Lahn et al. (2019). **(I1):** The algorithm maintains a 1-feasible transport plan, and, **(I2)** In each phase, the partial DFS step computes at least one augmenting path. Furthermore, at the end of the partial DFS, there is no augmenting path in the admissible graph.

Next, we show that any transport plan maintained by the algorithm will satisfy condition (C'). For $v \in A$, initially $y(v) = 0$. In any phase, suppose $\ell_v < \ell_t$. Then, the Hungarian Search updates the dual weights using condition (U1') which reduces the dual weight of v. Therefore, $y(v) \leq 0$. For $v \in B$, initially $y(v) = 0$. In any phase, suppose $\ell_v < \ell_t$. Then, the Hungarian Search updates the dual weights using condition (U2') which increases the dual weight of v. Therefore, $y(v) \geq 0$.

Next, we show that all free vertices of A have a dual weight of 0. The claim is true initially. During the course of the algorithm, any vertex $a \in A$ whose demand is met can no longer become free. Therefore, it is sufficient to argue that no free demand vertex experiences a dual adjustment. By construction, there is a directed edge from v to t with zero cost in $\mathcal{G}_\sigma$. Therefore, $\ell_t \leq \ell_v$ and the algorithm will not update the dual weight of v during the phase. As a result the algorithm maintains $y(v) = 0$ for every free demand vertex and (C') holds.

Finally, we show that all free vertices of B have a dual weight of $y_{\max}$. The claim is true initially. During the course of the algorithm, any vertex $b \in B$ whose supply is transported can no longer become free. Therefore, it is sufficient to argue that a free supply vertex experiences the largest increase in dual weights. By construction, there is a directed edge from s to v with zero cost in $\mathcal{G}_\sigma$. Therefore, $\ell_v = 0$ and the algorithm will increase the dual weight of v by the largest value ℓ_t during the phase. As a result the algorithm maintains $y(v) = y_{\max}$ for every free supply vertex and (C') holds.

Generating $\overline{\omega}$ and $D\overline{\omega}$: As shown in Lahn et al. (2019), the algorithm terminates in $q = O(1/\delta)$ phases. Let $\{\sigma_0, \sigma_1, \sigma_2, \ldots, \sigma_q\}$ be the transport plans where σ_{i-1} is the transport plan prior to the start of phase i. Let σ_q denote the transport plan at the end of phase q. Let α_i be the mass transported by σ_i. Let $y^i_{\max}$ denote the $y_{\max}$ value after phase i. Let p_i denote the point $(\alpha_i/\theta, w(\sigma_i))$. The approximate OT-profile is a function given by the sequence $\langle p_1, \ldots, p_q \rangle$ where every adjacent pair of points is connected by a line segment. The function $D\overline{\omega}$ is given as follows: $D\omega(0) = y^1_{\max}$. For every $\alpha \in (\alpha_{i-1}/\theta, \alpha_i/\theta]$, $D\overline{\omega}(\alpha) = y^i_{\max}$. Next, we show that the function $D\overline{\omega}$ satisfies an approximate version of the outlier lemma which we state below: Consider the mass at B and A after scaling the demands and supplies. Let w be the OT-cost of transporting mass (after scaling) from B^+ to A with respect to costs $c(\cdot, \cdot)$. We assume that for any $(a, b) \in A \times B^-$, the cost of the edge is Cw/ε for some constant $C > 4$.

Lemma 7.3. *Suppose $\delta\mathcal{S} \leq w/4$, then the $(\min\{\theta\alpha^*, \mathcal{S}\} - \varepsilon)$-partial transport plan generated by our approximation algorithm does not transport any mass from the outlier set B^-. Furthermore, $D\overline{\omega}(\alpha^* - \varepsilon) \leq (4w + \delta\mathcal{S})/\delta\varepsilon$ and $D\overline{\omega}(\alpha^* + \varepsilon) \geq 2Cw/\delta\varepsilon$.*

Thus, as long as the additive error is sufficiently small in comparison to w, we will be able to identify a jump in the first derivative function at α^*. Furthermore, we can mark the free vertices of $(\min\{\theta\alpha^*, \mathcal{S}\} - \varepsilon)$-partial transport plan generated by our algorithm as outliers.

Proof for the Approximate Outlier Detection Lemma: The proof of the approximate outlier detection lemma is very similar to the exact one.

We begin by providing an overview of the proof of the approximate outlier lemma. First, in Lemma 7.4 (whose proof is similar to Lemma 6.8, we show that the $(\min\{\theta\alpha^*, \mathcal{S}\} - \varepsilon)$-partial transport plan σ generated by our approximation algorithm for input sets A and B^+ is also an $(\min\{\theta\alpha^*, \mathcal{S}\} - \varepsilon)$-optimal partial transport plan generated by our algorithm for the sets A and B, and therefore, σ does not transport any mass from outlier points.

Next, in Lemma 7.5, we argue that the dual weights generated by our algorithm while computing $(\min\{\theta\alpha^*, \mathcal{S}\} - \varepsilon)$-partial transport plan σ is at most $y_{\max} \leq (4w + \delta\mathcal{S})/\delta\varepsilon$. This implies that $D\overline{\omega}(\alpha^* - \varepsilon) \leq (4w + \delta\mathcal{S})/\delta\varepsilon$. On the other hand, any transport plan σ' that transport a mass of $(\alpha^* + \varepsilon)\theta$ will also transport some mass from the outlier points B^-, i.e., there is some edge $(a, b) \in A \times B^-$ with $\sigma'(a, b) > 0$. From feasibility of σ', we have $y(a) + y(b) \geq \overline{w}(a, b)$ Since, for any $(a, b) \in A \times B^-$, its weight $\overline{w}(a, b) \geq 2Cw/\delta\varepsilon$ and since $y(a) \leq 0$ (from (C')), we conclude that $y_{\max} \geq y(b) \geq 2Cw/\delta\varepsilon$. By its definition $D\overline{\omega}(\alpha^* + \varepsilon) \geq y_{\max} \geq 2Cw/\delta\varepsilon$.

Lemma 7.4. *Let σ be the $(\min\{\theta\alpha^*, \mathcal{S}\} - \varepsilon)$-partial transport plan σ generated by our approximation algorithm for input sets A and B^+. Then, σ is also the $(\min\{\theta\alpha^*, \mathcal{S}\} - \varepsilon)$-partial transport plan generated by our algorithm with A and B as the input.*

Lemma 7.5. *Consider the $(\theta\alpha^* - \varepsilon)$-optimal partial transport σ between A and $B^{\mathcal{I}}$ computed by our exact algorithm. Let j be the phase where σ is obtained. Then, at the end of phase j, the largest dual weight $y^j_{\max} \leq 4(w + \delta\mathcal{S})/\delta\varepsilon$.*

Proof. Note that the total inlier mass (after scaling demands and supply) is $\theta\alpha^*$. Let $\overline{w}$ denote the optimal transport cost with respect to the costs $\overline{c}(\cdot,\cdot)$. Also, note that the optimal transport cost between A and $B^{\mathcal{I}}$ with respect to the scaled costs, denoted by $\overline{w}$ is at most $4w/\delta$, i.e., $\overline{w} \leq 4w/\delta$. Let σ be the $(\theta\alpha^* - \varepsilon)$-optimal partial transport between A and $B^{\mathcal{I}}$ as computed by our algorithm. Let σ^* be the optimal transport plan.

First, without loss of generality, we transform σ and σ^* so that σ remains a 1-feasible transport plan, σ^* remains a maximum transport plan and $w(\sigma^*) - w(\sigma)$ remains unchanged. Furthermore, this transformation guarantees that the dual weights for the 1-feasible transport plan σ is such that every edge (a, b) for which the optimal transport plan has a positive flow, i.e., $\sigma^*(a, b) > 0$, also satisfies equation 26. We present this transformation next.

If the dual weights for an edge (a, b) do not satisfy equation 26, then its flow $\sigma(a, b)$ is $\min\{\overline{s}_b, \overline{d}_a\}$. We reduce $\overline{d}_a$ and $\overline{s}_b$ by $\sigma^*(a, b)$ and also reduce the flow on the edge (a, b) in σ^* to be 0 and σ to $\min\{\overline{s}_b, \overline{d}_a\} - \sigma^*(a, b)$. The transformed σ continues to be 1-feasible transport plan and the transformed σ^* is a maximum transport plan for the new demands and supplies. Moreover, the difference in costs of σ and σ^* does not change due to this transformation and we are guaranteed that if the edge (a, b) has a positive flow with respect to σ^*, i.e., $\sigma^*(a, b) > 0$ then (a, b) will satisfy equation 26. We present the rest of the proof assuming that σ and σ^* are transformed.

For any point $a \in A$ (resp. $b \in B$), let x_a (resp. x_b) denote the deficit at a (resp. b) with respect to σ. Recollect that $\overline{d}_a = \sum_{b\in B} \sigma(a, b) + x_a$ and $\overline{s}_b = \sum_{a\in A} \sigma(a, b) + x_b$. Using this, we can write

$$\sum_{a\in A} \overline{d}_a y(a) + \sum_{b\in B} \nu_b y(b) = \sum_{(a,b)\in A\times B} \sigma(a, b)(y(a) + y(b)) + \sum_{a\in A} y(a)x_a + \sum_{b\in B} y(b)x_b.$$

From (C), we conclude that if $x_a > 0$, then $y(a) = 0$ and if $x_b > 0$, then $y(b) = y^j_{\max}$. Using this and the fact that if $\sigma(a, b) > 0$, then $y(a) + y(b) \geq \overline{c}(a, b)$, we get

$$\sum_{a\in A} \overline{d}_a y(a) + \sum_{b\in B} \overline{s}_b y(b) \geq \sum_{(a,b)\in A\times B} \sigma(a, b)\overline{c}(a, b) + y^j_{\max} \sum_{b\in B} x_b,$$

which can be rewritten as

$$\sum_{a\in A} \overline{d}_a y(a) + \sum_{b\in B} \overline{s}_b y(b) \geq \overline{w}(\sigma) + y_{\max}\varepsilon. \tag{32}$$

Let x^*_a be the deficit and excess with respect to σ^* (Since σ^* is a maximum transport plan, the surplus at each supply node remains 0). We can write

$$\sum_{a\in A} \mu_a y(a) + \sum_{b\in B} \nu_b y(b) = \sum_{(a,b)\in A\times B} \sigma'(a, b)(y(a) + y(b)) + \sum_{a\in A} y(a)x'_a.$$

Since σ is feasible, due to the transformation, for any edge (a, b) with $\sigma^*(a, b) > 0$, $y(a) + y(b) \leq \overline{c}(a, b) + 1$. Furthermore, from (C) every vertex $a \in A$ has $y(a) \leq 0$. Using these inequalities, we get

$$\sum_{a\in A} \overline{d}_a y(a) + \sum_{b\in B} \overline{s}_b y(b) \leq \sum_{(a,b)\in A\times B} \sigma^*(a, b)\overline{c}(a, b) + \mathcal{S},$$

or

$$\sum_{a\in A} \overline{d}_a y(a) + \sum_{b\in B} \overline{s}_b y(b) \leq \overline{w}(\sigma^*) + \mathcal{S}. \tag{33}$$

From equation 32 and equation 33, we conclude that $w(\sigma') + \varepsilon y^j_{\max} \leq \overline{w} + \mathcal{S}$, or $y^j_{\max} \leq (\overline{w} + \mathcal{S})/\varepsilon$. Plugging in $\overline{w} \leq 4w/\delta$, we get $y^j_{\max} \leq (4w + \varepsilon\mathcal{S})/\delta\varepsilon$

□

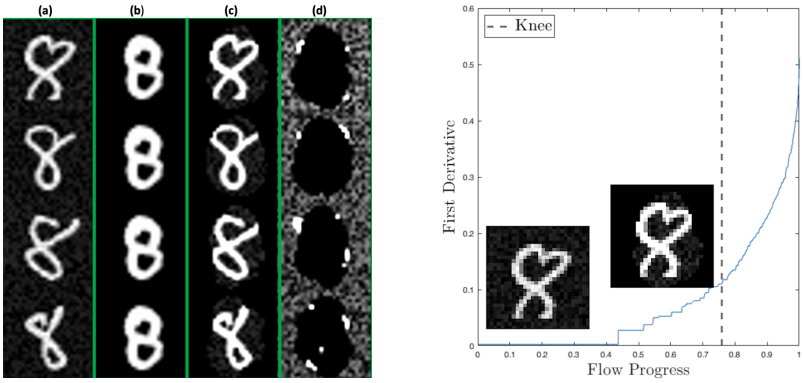

Figure 2: Noise removal from MNIST images

8 Additional Experimental Details

8.1 Noise Removal Experiment

We highlight the benefit of our automated outlier detection with the following example. In Figure 2 (Left), the column **a** corresponds to images that have a 30% white noise in the background and the column **b** corresponds a clean image of the digit 8 from the MNIST dataset. We compute an approximate OT-profile as well as approximate its first derivative for the noisy and clean images in each row using our method. We observe a sharp increase in the first derivative which marks the onset of outliers. To detect this point, we use the kneedle method Satopaa et al. (2011) which uses a spline based interpolation of the data, approximation of the curvature and heuristics to detect the knee in order to obtain the inlier (not containing noise) and the outlier (noise) distributions. The columns **c** and **d** correspond to these inlier and the outlier distributions (their mass is scaled to 1), respectively. In Figure 2 (Right), we show an example of the first derivative plot from OT-profile between the the first image from column (a) and the first image from column (b). The trimmed image (first image from column c) is shown at the detected knee point.

8.2 Details of Knee Detection Method for the Outlier Detection Experiment

In the case of outlier detection, we compute the approximate OT-profile and its first derivative using our method between the clean (μ) and noisy (ν) datasets. We smoothen the first derivative (y-axis) curve by using a moving average over a sliding window along flow progress (x-axis). The width of the sliding window is 0.01. Next, we flip the x and y axis of the first derivative curve (for example the one in Figure 2 (Right)), we see a curve that seems to obey the law of diminishing returns. This shows that matching outliers in the noisy data causes the cost to increase sharply after a point. We run the kneedle method (with the default sensitivity parameter of 1) to identify this knee point for the curve which is reported as the approximation to α^*.

8.3 Details of Jump Detection in the PU Learning Experiment

We use the OT-profile and automated jump detection method (described below) to identify an approximation $\hat{\pi}$ of the class prior π and then use $\hat{\pi}$-optimal partial transport to determine the positive unlabelled data. We refer to this approach as *OTP-wo-prior*.

The jump detection method contains two steps. Recollect that the first derivative is a step function (See the blue curve in Figure 3(left)). In the first step, to smoothen the first derivative, we interpolate between the start points of step in the first derivative (See red curve in Figure 3(left)). In the second step, for every interval of width 0.01, we calculate the ratio of the first derivative value at the right end of the interval to the first derivative value at the left end of the interval. Figure 3 (right) plots these ratio for all the intervals. Finally, we detect the leftmost peak that has a prominence larger than 0.1 (We assume that the values along both axis are normalized to 1).

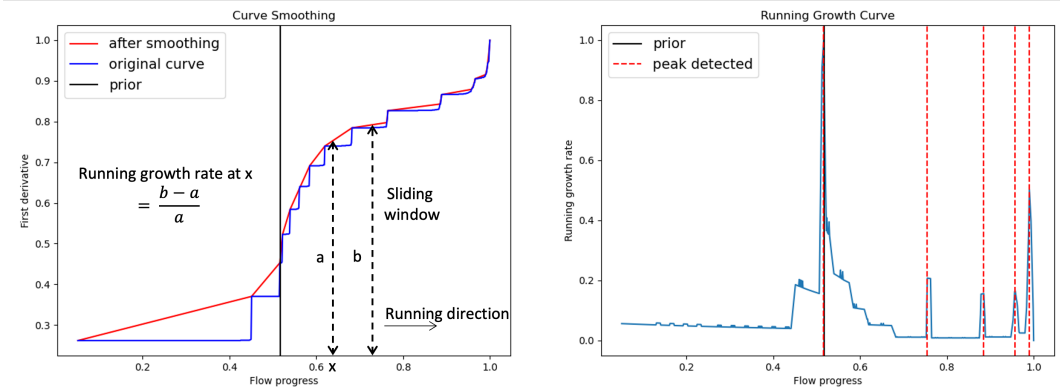

Figure 3: Prior detection from OT-profile in PU-learning

8.4 Outlier Detection On Synthetic Data

Experiment with Synthetic Dataset: In this experiment, μ is a 'clean' set of $n = 10000$ samples drawn from a 2-dimensional gaussian distribution centered at $(0, 0)$ and with variance of 1 across all dimensions. The 'contaminated' set ν consists of n samples drawn from the distribution $(1-\varepsilon)\mathcal{N}_0 + \varepsilon\mathcal{N}_\eta$, where $\mathcal{N}_\eta$ is the noise distribution with mean (η, η) and variance 1. ε is the noise contamination rate. We assign a mass of $1/n$ to each sample point. We consider $\varepsilon \in \{0.2, 0.3\}$ and for each ε, we conduct three experiments with the center of the outlier distribution η set to 2, 1 and 0.5. For each choice of η and ε, we report (averaged over 10 experiments), in Table 3, the L_1-distance of the predicted mean to the actual mean of the noise distribution (η, η) (in Row 1). We also report the inlier mass, i.e., $(1-\varepsilon)$, as predicted by the 'kneedle' method, in Row 2. We observe that our prediction for the inlier mass become less accurate as the centers of the contamination distribution approaches the center of the clean distribution.

Table 3: Outlier Detection for Synthetic Data. Row 1: Estimated mean of the noise distribution. Row 2: Predicted inlier mass. Results shown for noise distributions $\mathcal{N}_2$(Left), $\mathcal{N}_1$ (Mid), and $\mathcal{N}_{0.5}$ (Right).

$\varepsilon = 0.3$	$\varepsilon = 0.2$
$0.03 \pm 0.02.$	0.04 ± 0.02
$0.80 \pm 0.00.$	0.86 ± 0.01

$\varepsilon = 0.3$	$\varepsilon = 0.2$
$0.04 \pm 0.02.$	0.04 ± 0.02
$0.87 \pm 0.00.$	0.91 ± 0.00

$\varepsilon = 0.3$	$\varepsilon = 0.2$
$0.06 \pm 0.02.$	0.06 ± 0.02
$0.94 \pm 0.00.$	0.95 ± 0.00

One more value of ε and one more noise distribution, which are shown in Table 4, 5 and 6. Same as Row 1 of Table 3, Table 5 contains the data of the L_1-distance of the predicted mean to the actual mean of the noise distribution (η, η). And Table 6, like Row 2 of Table 3, reports the inlier mass, i.e., $(1-\varepsilon)$, as predicted by the 'kneedle' method. Moreover, we show the outlier mean estimation in Table 4.

Table 4: Outlier Mean Estimation for Synthetic Data, $n = 10k$

Outlier Distribution (n)	$\varepsilon = 0.3$	$\varepsilon = 0.25$	$\varepsilon = 0.2$
$\mathcal{N}(2, \mathbf{1_2})$	2.005 ± 0.021	2.005 ± 0.025	2.007 ± 0.028
$\mathcal{N}(1, \mathbf{1_2})$	1.006 ± 0.029	1.007 ± 0.031	1.007 ± 0.033
$\mathcal{N}(0.5, \mathbf{1_2})$	0.509 ± 0.04	0.505 ± 0.045	0.506 ± 0.047
$\mathcal{N}(0.1, \mathbf{1_2})$	0.078 ± 0.085	0.074 ± 0.083	0.075 ± 0.081

8.5 Additional Details of Outlier Detection on Real Datasets

For the outlier detection experiment on MNIST data, we present the count of misclassified digits as well as show examples of misclassified digits. Figure 4 shows a visual sample of the miss-classified

Table 5: Outlier Mean Estimation Error for Synthetic Data, $n = 10k$

Outlier Distribution (n)	$\varepsilon = 0.3$	$\varepsilon = 0.25$	$\varepsilon = 0.2$
$\mathcal{N}(2, \mathbf{1_2})$	0.026 ± 0.015	0.032 ± 0.018	0.037 ± 0.018
$\mathcal{N}(1, \mathbf{1_2})$	0.039 ± 0.017	0.04 ± 0.02	0.043 ± 0.02
$\mathcal{N}(0.5, \mathbf{1_2})$	0.055 ± 0.02	0.06 ± 0.023	0.062 ± 0.025
$\mathcal{N}(0.1, \mathbf{1_2})$	0.107 ± 0.062	0.108 ± 0.061	0.105 ± 0.058

Table 6: Check Point Detected for Synthetic Data, $n = 10k$

Outlier Distribution (n)	$\varepsilon = 0.3$	$\varepsilon = 0.25$	$\varepsilon = 0.2$
$\mathcal{N}(2, \mathbf{1_2})$	0.799 ± 0.004	0.835 ± 0.003	0.861 ± 0.014
$\mathcal{N}(1, \mathbf{1_2})$	0.867 ± 0.003	0.887 ± 0.003	0.907 ± 0.002
$\mathcal{N}(0.5, \mathbf{1_2})$	0.937 ± 0.002	0.945 ± 0.002	0.953 ± 0.002
$\mathcal{N}(0.1, \mathbf{1_2})$	0.984 ± 0.001	0.984 ± 0.001	0.985 ± 0.001

MNIST images for ROBOT and OT-profile approaches. Table 7 gives the errors made on a sample of $n = 2000$ by ROBOT and OT-Profile for each digit.

Table 7: Number of errors made by ROBOT and our method for $n = 2000$, $\varepsilon = 0.2$. 0-4 are inliers detected wrongly as outliers and 5-9 are outliers classified as inliers

Digit	ROBOT	Ours
0	2	21
1	0	4
2	1	59
3	7	56
4	5	48
5	66	12
6	64	11
7	63	19
8	78	32
9	69	50

8.6 Example First Derivative curves with knee points for Synthetic Data

In this section, we show the first derivative plots (Figure 5, 6 and 7) from OT-profile between the synthetic data as described in the experiment section. Here the x-axis is the progress of optimal transport flow. We use the data from cumulative flow at each iteration divided by the total flow. Y-axis is the first derivative of transport cost. Here we used the data from the dual weight of the start vertex of the augmentation path discovered in each iteration. And, like Figure 1, the detected knee point is shown in the graph. For each synthetic data setup, we show one example out of ten experiments.

8.7 Example First Derivative curves with knee points for Real Data

In this section, we show the first derivative plots (Figure 8) from OT-profile between the real data (MNIST) as described in the experiment section. Like previous section, the plot is drawn based on the same methodology. For each real data experiment setup, we show one example out of ten experiments.

8.8 Additional Result for PU Learning - Standard Deviation

Due to the page limit, we report the standard deviation of PU learning accuracy rate and execution time in the following table 8.8.

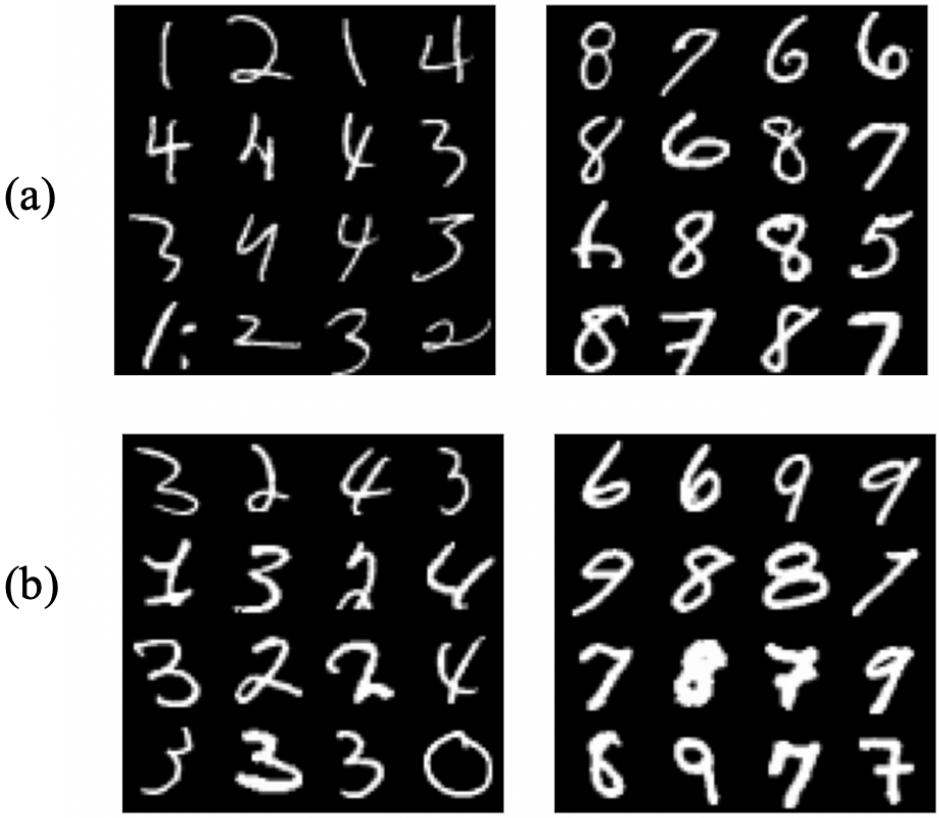

Figure 4: Outlier Detection - Comparison with ROBOT. (a) ROBOT : left - Inliers detected as Outliers, right - Outliers detected as Inliers. (b) Ours : left - Inliers detected as Outliers, right - Outliers detected as Inliers.

Table 8: Standard Deviation of PU Learning Accuracy Rates (up) and Execution Time(down)

dataset	p-W	p-GW	OTP-w-prior	OTP-wo-prior	$\hat{\pi}$
mushrooms	0.8	1.0	0.3	0.1	0.001625
shuttle	1.1	1.4	0.9	0.8	0.007842
pageblocks	0.9	1.2	0.7	1.1	0.030288
usps	0.5	1.1	0.5	1.7	0.017550
connect-4	2.0	1.7	1.5	1.0	0.042889
spambase	1.8	1.5	1.3	12.7	0.343822
mnist	0.004	0.004	0.003	0.003	0.007972
colored mnist	0.004	81.835	0.002	0.002	0.010238

dataset	p-W	p-GW	OTP-w-prior	OTP-wo-prior
mushrooms	0.185	0.428	0.016	0.248
shuttle	0.027	1.396	0.012	0.079
pageblocks	0.197	2.365	0.022	0.024
usps	0.187	2.145	0.210	0.214
connect-4	0.157	1.087	0.230	0.245
spambase	0.100	0.766	0.064	0.047
mnist	0.000	1.063	0.050	0.255
colored mnist	0.027	1.396	0.012	0.079

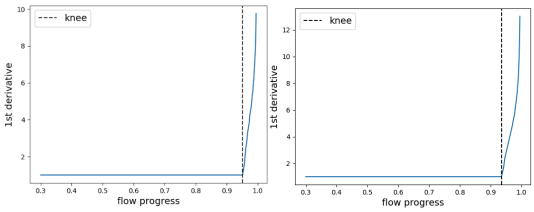

Figure 5: Noise Detection from Synthetic Data, Noise Distribution $\mathcal{N}_{0.5}$, Left: $\mu = 0.2$, Right: $\mu = 0.3$

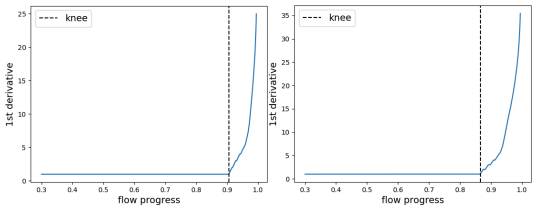

Figure 6: Noise Detection from Synthetic Data, Noise Distribution $\mathcal{N}_1$, Left: $\mu = 0.2$, Right: $\mu = 0.3$

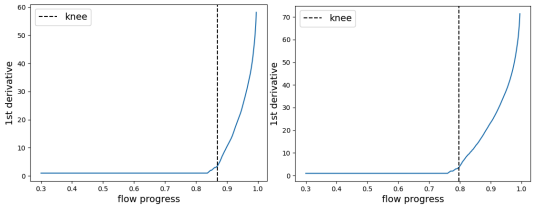

Figure 7: Noise Detection from Synthetic Data, Noise Distribution $\mathcal{N}_2$, Left: $\mu = 0.2$, Right: $\mu = 0.3$

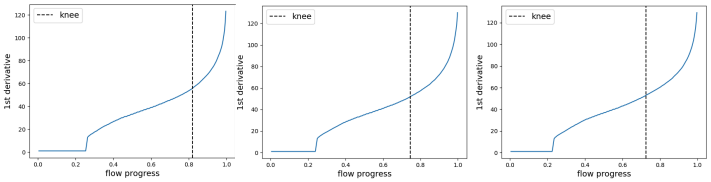

Figure 8: Noise Detection from Real Data (MNIST), Up Left: $\mu = 0.2$, Up Right: $\mu = 0.25$, Down: $\mu = 0.3$