This Appendix is organized as follows. In Section 6, we will provide the missing proofs and details of Section 2. In Section 7, we present the missing details and proofs of Section 3. In Section 8, we present the details of how LMR-algorithm can be modified to compute an approximate OT-profile and also state and prove the approximate outlier lemma. In Section 9, we present the details of our experimental result as well as report some additional experiments.

6 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 $\sigma(a,b) = \min\{\mu_a, \nu_b\}$. □

Lemma 6.2. *Given a feasible α-partial transport plans σ and σ' with $w(\sigma') < w(\sigma)$, we can transform σ and σ' so that σ remains feasible, σ and σ' transport the same mass, the surplus (resp. deficit) at each node with respect to σ (resp. σ') stays the same, 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.*

Proof. 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. $a \in A$). □

Proof of Lemma 2.1: For the sake of contradiction, let σ' be another partial transport plan carrying a mass of α such that $w(\sigma') < w(\sigma)$. Assume that σ and σ' have been transformed as described in Lemma 6.2.

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

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

From our initial transformation of Lemma 6.2, any edge (a,b) with $\sigma'(a,b) > 0$ also satisfies equation 1, i.e., $y(a) + y(b) \le c(a,b)$. Thus,

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

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

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

or

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

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 $w(\sigma') \ge w(\sigma)$ 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 $\sigma(a',b') + k \le \sigma(a',b') + (\min\{\mu_a,\nu_b\} - \sigma(a',b')) \le \min\{\mu_a,\nu_b\}$.

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., $r_P \le \mu_a - \sum_{b'\in B} \sigma(a,b')$. Using the fact that $k \le r_P$, we can rewrite this as

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

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_b. By our choice, r_P is at most the surplus at b, i.e., $r_P \le \nu_b - \sum_{a'\in A} \sigma(a',b)$. Using the fact that $k \le r_P$, we can rewrite this as

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

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

7 DETAILS OF SECTION 3

In this section, we present a pseudocode of our exact algorithm as well as the missing proofs from Section 3.

7.1 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 7.1, 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.

Algorithm 1 Our Exact Algorithm

Input: μ (with support A, mass U), ν (with support B, mass S), $\forall(a,b) \in A \times B, c(a,b)$.
Initialization:
Set $\alpha \leftarrow 0$; $\forall(a,b) \in A \times B, \sigma(a,b) \leftarrow 0$; $\forall v \in A \cup B, y(v) \leftarrow 0$
while $\alpha < S$ **do** ▷ Phases
  First Step (Hungarian Search):
  Execute Dijkstra's shortest path algorithm on the augmented residual network $\mathcal{G}_\sigma$
  $\forall v \in A \cup B$, get ℓ_v as the shortest distances from s to v returned by Dijkstra's algorithm
  for $\forall v \in A \cup B$ with $\ell_v < \ell_t$ **do** ▷ Recollect t is the sink in $\mathcal{G}_\sigma$
    if $v \in A$ **then**
      $y(v) \leftarrow y(v) - \ell_t + \ell_v$
    end if
    if $v \in B$ **then**
      $y(v) \leftarrow y(v) + \ell_t - \ell_v$
    end if
  end for
  Second Step:
  Execute DFS on the admissible graph $\overline{\mathcal{A}}_\sigma$ to find an augmenting path P
  Augment σ along P by a mass of r_P
  $\alpha \leftarrow \alpha + r_P$
end while
return σ

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 7.2) 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 7.3, 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 7.1. *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 7.2. *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 7.3. *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. □

7.2 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, $y^i_{\max} = y(b) + \ell_t - \ell_b = y^{i-1}_{\max} + \ell_t$.

7.3 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 $w(\sigma_{i-1}) + (\alpha - \alpha_{i-1})\Phi(P_i) = \sum_{j=1}^{i-1} r_j y^j_{\max} + (\alpha - \alpha_{i-1}) y^i_{\max}$.

7.4 Proof of Lemma 3.5

If all optimal partial transports are unique, then from Lemma 7.4, there is no augmenting path of admissible edges after any phase i. Since there is no augmenting path in the admissible graph, from Lemma 7.5, 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 7.4. *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 7.5. *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. □

7.5 Proof of Lemma 1.1

Now, we 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).

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 7.6, 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 7.7, 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 7.6. *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 $v \in B^+ \cup A$, $|y(v)| \le w/\varepsilon$.*

Lemma 7.7. *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 7.6 and 7.7 already establish that $D\omega(\alpha^* - \varepsilon) \le w/\varepsilon$. To show that $D\omega(\alpha^* + \varepsilon) \ge Cw/\varepsilon$, 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 $c(a, b) \ge Cw/\varepsilon$. Since $\sigma'(a, b) > 0$, by equation 2, $y'(b) + y'(a) \ge c(a, b) \ge Cw/\varepsilon$. Since, from (C), $y'(a) \le 0$, we have $y'_{\max} \ge y'(b) \ge Cw/\varepsilon$. From Lemma 3.3, it follows that $D\omega(\alpha^* + \varepsilon) \ge Cw/\varepsilon$.

7.5.1 Proof of Lemma 7.6

Let σ^* be the optimal transport plan between μ and ν^+. Recollect that w is the cost of σ^*. Also, recollect that ν^+ is simply the mass distribution obtained by removing points of B^- from ν. Therefore, ν^+ is simply the mass distribution where each $b \in B^+$ has a mass of ν_b.

We apply the transformation of Lemma 6.2 on σ and σ^* and 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 $y(a) + y(b) \ge c(a, b)$ (from equation 2), we get

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

which can be rewritten as

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

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

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

For any $a \in A$, let x'_a be the deficit with respect to σ^*. Since σ^* transports all supplies from ν^+, 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., $y(a) + y(b) \le c(a,b)$. Thus,

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

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

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

or

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

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

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

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

7.5.2 Proof of Lemma 7.7

Recollect that σ is the $(\alpha^* - \varepsilon)$-optimal partial transport plan generated by our algorithm when executed on the distributions μ and ν^+. 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, the dual weights associated with σ, i.e., $y(\cdot)$, will also be the dual weights before the execution of Step 2 of phase i and σ' and $y(\cdot)$ will form a feasible transport plan. Let $\beta < \alpha^* - \varepsilon$ be the mass transported by σ'.

We begin by assigning a dual weight to every vertex $b \in B^-$, $y(b) \leftarrow y_{\max}$. By Lemma 7.6,

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

We show that every forward edge $\overrightarrow{ba}$ remains feasible.[3]

By the outlier assumption, every edge $(a,b) \in A \times B^-$, $c(a,b) > w/\varepsilon$. Since our algorithm maintains the dual weight $y(a) \le 0$, it follows from equation 22 that $y(b) \le w/\varepsilon$. Therefore, for every edge $(a,b) \in A \times B^-$, the sum $y(a) + y(b) \le w/\varepsilon \le c(a,b)$.

After introducing the points of B^- and assigning them dual weights, note that 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.

Next, we bound the first derivative at $\alpha^* - \varepsilon$, i.e., $D\omega(\alpha^* - \varepsilon)$. Observe that both 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 $D\omega(\alpha^* - \varepsilon) = y_{\max} \le w/\varepsilon$ (From Lemma 7.6).

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

8 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.

8.1 SCALING DEMANDS AND SUPPLIES

The LMR-algorithm has three parts. In this section, we describe the scale-and-round step (Part 1) of the LMR algorithm that converts the demands and supplies to integers. Then, in Part 2, the LMR-algorithm then finds an approximate solution to the transformed problem in $O(\frac{n^2}{\delta} + \frac{n}{\delta^2})$ time. Finally, in Part 3, this solution is mapped back to the original demands and supplies. The total loss in accuracy in the cost due to this transformation (Part 1 and 3) is at most $S\delta/2$. The notation and the presentation of the algorithm will be similar to how it was done in Lahn et al. (2019). This will allow us to use several properties of the LMR-algorithm that was derived in Lahn et al. (2019).

We describe the details of Part 1. Set the scaling parameter $\theta = \frac{4n}{S\delta}$. Let $\mathcal{I}$ be the input instance for the optimal transport problem. Recollect, each supply location $b \in B$ has a mass of ν_b and each demand location $a \in A$ has a demand of μ_a. In this step, a new input instance $\mathcal{I}'$ is created where the demands are scaled and rounded up, i.e., at each node $a \in A$ the demand is set to $\overline{d}_a = \lceil \mu_a \theta \rceil$ and the supplies 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

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

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 shown in Lahn et al. (2019), the transport plan σ is neither a feasible plan nor an α-partial transport plan for $\mathcal{I}$. Nonetheless, it can be converted into one with an additional increase in the transport cost of at most $2n/\theta$.

The cost of such a transformed α-partial transport plan is

$$w(\sigma) \leq w(\sigma')/\theta + \frac{2n}{\theta} \leq w(\sigma')/\theta + S\delta/2. \tag{24}$$

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 8.1 (whose proof is in Lahn et al. (2019)), it can be shown that $w(\sigma'_{\mathrm{OPT}}) \leq \alpha w(\sigma^*)$. In the Section 8.2, 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 8.1(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 24, 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 8.1. *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 8.2 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, Part 3 of the algorithm converts the OT profile into a δ-approximate OT profile $\overline{\omega}$ for $\mathcal{I}$ by simply setting $\overline{\omega}(\alpha) = \omega'(\alpha)/\theta + S\delta/2$.

8.2 PART 2 OF THE LMR-ALGORITHM

Given 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. The LMR-algorithm can be modified to produce an approximate OT-profile in $O(n^2/\delta + n/\delta^2)$ time.

Scaling Costs: 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 $c(\cdot, \cdot)$ and $\overline{w}(\sigma)$ denotes the cost of any transport plan with respect to the $\overline{c}(\cdot, \cdot)$.

1-feasible transport plan: 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 $b \in B$,

$$y(a) + y(b) \leq \overline{c}(a, b) + 1 \qquad \text{if } \sigma(a, b) < \min\{\overline{s}_b, \overline{d}_a\} \tag{25}$$
$$y(a) + y(b) \geq \overline{c}(a, b) \qquad \text{if } \sigma(a, b) > 0. \tag{26}$$

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 25 is a relaxation of equation 1. 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 8.2, 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 8.2. *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 $w(\sigma') + \mathcal{S}\delta/2 < w(\sigma)$.

We use a transformation similar to the one described in Lemma 6.2, and present the rest of the proof assuming that σ and σ' are transformed with the following property. After the transformation, 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 25.

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

$$\sum_{a\in A} \overline{d}_a y(a) + \sum_{b\in B} \overline{s}_b y(b) \geq \overline{w}(\sigma) + y_{\max}(\mathcal{S} - \overline{\alpha}). \tag{27}$$

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

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

From equation 27 and equation 28, 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 25 and redefine the slack on any edge between a and b in the residual network as

$$s(a,b) = \overline{c}(a,b) + 1 - y(a) - y(b) \qquad \text{if } (a,b) \text{ is a forward edge,} \tag{29}$$

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

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.

We modify the residual network $\overline{G}_\sigma$ to create a graph $\mathcal{G}_\sigma$ as follows: Add two additional vertices s and t. Add edges (with 0 weight) directed from s to every free supply node and add edges (with 0 weight) from every free demand vertex to t. The weight of every other edge (a,b) of the residual network is set to its slack $s(a,b)$ based on its direction.

8.2.1 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., $y(v) = 0$. The LMR-algorithm executes in *phases* and terminates when σ transports all the supplies. Within each phase there are two *steps*. These steps are very similar to those in the exact algorithm.

First step (Hungarian Search): To conduct a Hungarian Search, we execute Dijkstra's shortest path algorithm from s in $\mathcal{G}_\sigma$. For any vertex $v \in A \cup B$, let ℓ_v be the shortest path from s to v in $\mathcal{G}_\sigma$. 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 $v \in A$, $y(v) \leftarrow y(v) - \ell_t + \ell_v$, **(U2'):** Otherwise, if $v \in B$, $y(v) \leftarrow y(v) + \ell_t - \ell_v$.

This completes the first step. The second step of the algorithm finds 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').

- We begin by showing that $y(v) \leq 0$ for all $v \in A$. 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, $y(v) \leq 0$.
- Next, we show that for every $v \in B$, $y(v) \geq 0$. 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 8.3. *Suppose $\delta\mathcal{S} \le w/4$, 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, $D\overline{\omega}(\alpha^* - \varepsilon) \le (4w + \delta\mathcal{S})/\delta\varepsilon$ and $D\overline{\omega}(\alpha^* + \varepsilon) \ge 2Cw/\delta\varepsilon$.*

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 8.4 (whose proof is similar to Lemma 7.7, 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 8.5, 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 $y_{\max} \le (4w + \delta\mathcal{S})/\delta\varepsilon$. This implies that $D\overline{\omega}(\alpha^* - \varepsilon) \le (4w + \delta\mathcal{S})/\delta\varepsilon$. 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 $y(a) + y(b) \ge \overline{w}(a, b)$ Since, for any $(a, b) \in A \times B^-$, its weight $\overline{w}(a, b) \ge 2Cw/\delta\varepsilon$ and since $y(a) \le 0$ (from (C')), we conclude that $y_{\max} \ge y(b) \ge 2Cw/\delta\varepsilon$. By its definition $D\overline{\omega}(\alpha^* + \varepsilon) \ge y_{\max} \ge 2Cw/\delta\varepsilon$.

Lemma 8.4. *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 8.5. *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 $y^j_{\max} \le 4(w + \delta\mathcal{S})/\delta\varepsilon$.*

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., $\overline{w} \le 4w/\delta$. 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 25. We present this transformation next.

If the dual weights for an edge (a, b) do not satisfy equation 25, 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 25. 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

$$\sum_{a\in A} \overline{d}_a y(a) + \sum_{b\in B} \overline{s}_b y(b) \geq \overline{w}(\sigma) + y_{\max}\varepsilon. \tag{31}$$

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

$$\sum_{a\in A} \overline{d}_a y(a) + \sum_{b\in B} \overline{s}_b y(b) \leq \overline{w}(\sigma^*) + \mathcal{S}. \tag{32}$$

From equation 31 and equation 32, 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 $y^j_{\max} \leq (4w + \varepsilon\mathcal{S})/\delta\varepsilon$

□

9 Additional Experimental Details

9.1 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.

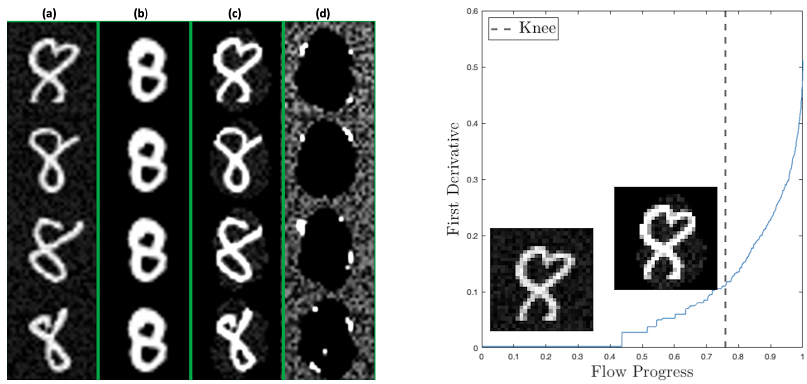

Figure 2: Noise removal from MNIST images

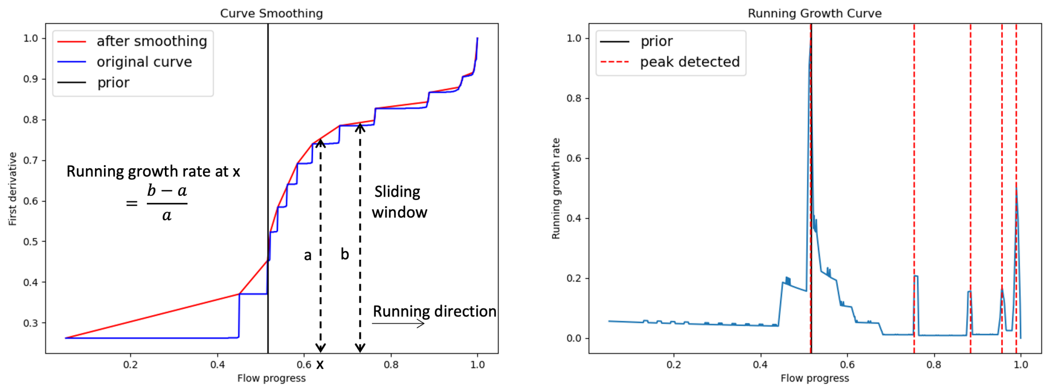

Figure 3: Prior detection from OT-profile in PU-learning

9.2 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^*.

9.3 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).

9.4 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

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

9.5 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 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.

9.6 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

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

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

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.

9.7 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.

9.8 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 9.8.

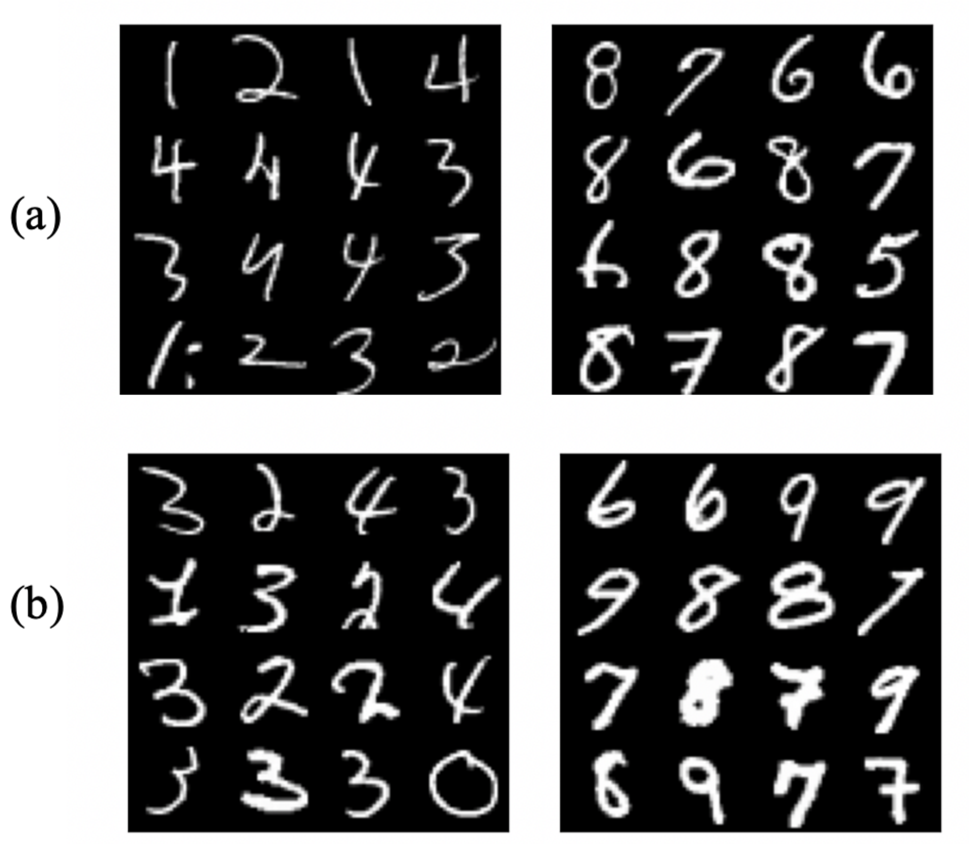

Figure 4: Outlier Detection - Comparison with ROBOT. (a) ROBOT : left - Inliers detected as Outliers, right - Outliers detected as Inliers. (b) Ours : left - Inliers detected as Outliers, right - Outliers detected as Inliers.

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	1.3	0.343822
mnist	0.004	0.004	0.003	0.003	0.007972
colored mnist	0.004	0.008	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

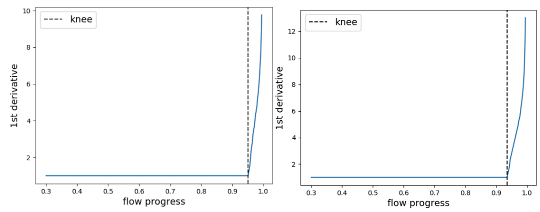

Figure 5: Noise Detection from Synthetic Data, Noise Distribution $\mathcal{N}_{0.5}$, Left: $\mu = 0.2$, Right: $\mu = 0.3$

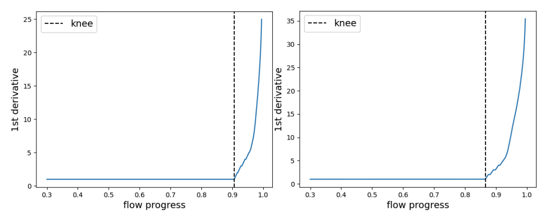

Figure 6: Noise Detection from Synthetic Data, Noise Distribution $\mathcal{N}_1$, Left: $\mu = 0.2$, Right: $\mu = 0.3$

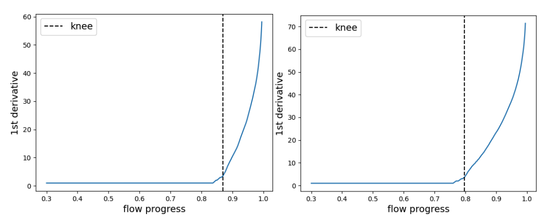

Figure 7: Noise Detection from Synthetic Data, Noise Distribution $\mathcal{N}_2$, Left: $\mu = 0.2$, Right: $\mu = 0.3$

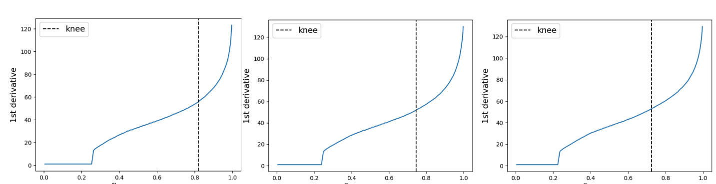

Figure 8: Noise Detection from Real Data (MNIST), Up Left: $\mu = 0.2$, Up Right: $\mu = 0.25$, Down: $\mu = 0.3$