Faster Partial Optimal Transport Computation with Combinatorial and Sparse Solvers

***Abstract*—This paper studies the Partial Optimal Transport (POT) problem between two unbalanced measures and its applications in various AI tasks like Domain Adaptation or PU learning. Motivated by the sparsity of transport plans, we first theoretically investigate the complexity of approximating the POT problem with quadratic regularization using the Adaptive Primal-Dual Accelerated Gradient Descent (APDAGD) algorithm. This results in an ε-approximate solution to the POT problem in $\mathcal{O}(n^{2.5}/\varepsilon)$, which is better in ε than feasible Sinkhorn. Additionally, we also propose a combinatorial framework to solve POT, which achieves the computation complexity of $\mathcal{O}(n^2/\varepsilon^2)$, making it a more scalable option. Lastly, We demonstrate the robustness of the proposed algorithms and the results' sparsity through real applications where two marginal distributions are unbalanced in tasks such as domain adaptation, PU learning or color transfer.**

I. Introduction

Optimal Transport (OT) [1, 2], which finds a minimum-cost matching between two balanced measures, is a well-studied topic in mathematics and control theory topics such as mean field games in [3, 4], Schrödinger bridges [5, 6] and neural ODEs [7]. However, OT's strict marginal conditions causes the map to be very sensitive to outliers [8, 9], as well as its practicality is hindered in tasks with unbalanced masses such as neuroimaging data averaging [10] and image classification [11, 12]. And so, OT variants such as the Partial Optimal Transport (POT), which only send a fraction of the mass, has been proposed and is gaining increasing significance. To solve this problem, several computational methods have been proposed. [13] is one of the first of such methods but fails to provide any convergence guarantee, while [14] proposed an adaptation of the Sinkhorn algorithm, but this fails to obtain a feasible solution. This infeasibility was later rectified in [15] where a feasible version of the Sinkhorn algorithm with a complexity of $O(n^2/\varepsilon^4)$ is proposed along with two gradient-based methods, one of which achieves the state-of-the-art theoretical complexity $O(n^2/\varepsilon)$ but has a poor empirical performance, while the other is computationally efficient in practice with a runtime complexity of $O(n^{2.5}/\varepsilon)$, which is less favorable in n compares to the existing algorithms. And so, this motivates us to explore computationally efficient algorithms that achieve an $O(n^2)$ scaling to combat the well-known curse of dimensionality issue in OT.

In addition to common gradient-based techniques, the discrete nature of the OT formulation has led to the consideration of combinatorial approaches, which have both classical and recent results. From Orlin et al. [16] with network simplex algorithms to Lahn et. al utilizing graph and matching approaches [17, 18], these methods leverage discrete settings in tasks such as image matching [19] and point cloud registration [20]. This motivates us to explore the possibility of making a POT solver from similar ideas.

We summarize our contributions as followed:

- We propose a combinatorial solver for POT as a generalization of Lahn et al.'s work [18] by combining it with the extended OT formulation [21], which achieve a theoretical time complexity of $\mathcal{O}(n^2/\varepsilon^2)$, and is also parallelizable.
- Motivated by the target of obtaining a sparse transport plan in OT using a quadratic regularizer as derived in [22], we derive the dual formulation of the quadratic regularized POT problem and then develope a sparse POT solver based on the APDAGD algorithm, which returns an ε-accurate feasible solution in $\mathcal{O}(n^{2.5}/\varepsilon)$ time.
- We conduct extensive experiments on two practical machine learning tasks—Domain Adaptation and Positive-Unlabeled (PU) Learning-to evaluate the performance of our algorithms.
- For Domain Adaptation, we compare two groups of methods: regularized and non-regularized. In each group, we experiment on the "moon" dataset, measuring the time taken, total cost, and sparsity of the transport plan for each method. Additionally, we train a support vector machine on the predicted target dataset and test it on the original source dataset to evaluate accuracy.
- For PU Learning, we evaluate our suite of POT algorithms on 8 benchmark datasets across 3 random seeds following the experimental setup from [21], covering both SCAR (*Selected Completely at Random*) and SAR (*Selected at Random*) labeling assumptions. Our results show that APDAGD and QAPDAGD consistently outperform Sinkhorn, Linear Programming, and QPdual on 7 out of 8 evaluated datasets.

II. Preliminaries

A. Notation

The set of first n positive integers is represented as $[n]$, while the set of non-negative real numbers is denoted by $\mathbb{R}_+$. We use bold capital letters, such as $\mathbf{A}$, to denote matrices and bold lowercase letters, such as $\mathbf{x}$, to represent vectors. Given an $m \times n$ matrix $\mathbf{X}$, the vectorization operation $\text{vec}(\mathbf{X})$ forms an (mn)-dimensional column vector by stacking the rows of $\mathbf{X}$ in sequence. Element-wise matrix operations are denoted using $\odot$ for multiplication and $\oslash$ for division.

For both vectors and matrices, the ℓ_p-norm is denoted as $\|\cdot\|_p$ for $1 \leq p \leq \infty$. The operator norm $\|\cdot\|_{p\to q}$ of a matrix is given by

$$\|\mathbf{A}\|_{p\to q} = \sup_{\|\mathbf{x}\|_p=1} \|\mathbf{A}\mathbf{x}\|_q.$$

A special case arises when $q = 2$, where $\|\mathbf{A}\|_{1\to q}$ corresponds to the largest ℓ_q-norm among the columns of $\mathbf{A}$. Additionally,

we define $\|\mathbf{A}\|_{\max}$ and $\|\mathbf{A}\|_{\min}$ as the maximum and minimum absolute values of the elements in $\mathbf{A}$, respectively.

The n-dimensional zero vector and all-ones vector are denoted as $\mathbf{0}_n$ and $\mathbf{1}_n$, respectively. Similarly, the $m \times n$ zero matrix is represented as $\mathbf{0}_{m,n}$. The probability simplex of dimension $n-1$ is formally defined as

$$\Delta_n = \{\mathbf{v} \in \mathbb{R}_+^n : \mathbf{v}^\top \mathbf{1}_n = 1\}.$$

B. Partial Optimal Transport Formulation

Let $\mathbf{r}$ and $\mathbf{c}$ be two discrete distributions in $\mathbb{R}_+^n$ that may have unequal total masses. The Partial Optimal Transport (POT) problem seeks to determine a transport plan $\mathbf{X} \in \mathbb{R}_+^{n\times n}$ that minimizes the cost of transferring mass from $\mathbf{r}$ to $\mathbf{c}$. Since the total masses of $\mathbf{r}$ and $\mathbf{c}$ may differ, the amount of mass transported, denoted as s, must satisfy the condition

$$0 \le s \le \min\{\|\mathbf{r}\|_1, \|\mathbf{c}\|_1\}.$$

The POT problem can thus be formally expressed as finding

$$\min_{\mathbf{X}\in\mathscr{U}(\mathbf{r},\mathbf{c},s)} \langle \mathbf{C}, \mathbf{X}\rangle \tag{1}$$

with the feasible set

$$\mathscr{U}(\mathbf{r},\mathbf{c},s) = \{\mathbf{X} \in \mathbb{R}_+^{n\times n} : \mathbf{X}\mathbf{1} \le \mathbf{r}, \mathbf{X}^\top \mathbf{1} \le \mathbf{c}, \mathbf{1}^\top \mathbf{X}\mathbf{1} = s\}$$

To increase flexibility using the inequalities constraints, we introduce auxiliary vectors $\mathbf{p}, \mathbf{q} \in \mathbb{R}_+^n$ such that:

$$\mathbf{X}\mathbf{1} + \mathbf{p} = \mathbf{r}$$
$$\mathbf{X}^\top \mathbf{1} + \mathbf{q} = \mathbf{c}$$

Thus, the initial POT problem can be rewritten as:

$$\min_{\mathbf{X}\in\mathbb{R}_+^{n\times n};\mathbf{p},\mathbf{q}\in\mathbb{R}_+^n} \langle \mathbf{C}, \mathbf{X}\rangle \tag{2}$$

$$\text{s.t} \quad \mathbf{X}\mathbf{1} + \mathbf{p} = \mathbf{r}, \mathbf{X}^\top \mathbf{1} + \mathbf{q} = \mathbf{c}, \mathbf{1}^\top \mathbf{X}\mathbf{1} = s \tag{3}$$

Definition 1: A transport plan $\mathbf{X}'$ satisfies (3) such that $\langle \mathbf{C}, \mathbf{X}'\rangle - \min_{\mathbf{X}\in\mathscr{U}(\mathbf{r},\mathbf{s},s)} \langle \mathbf{C}, \mathbf{X}\rangle \le \varepsilon$ is called an $\varepsilon-$approximate plan.

We can also combine the linear conditions into a single system, leading to the following equivalent expression:

$$\min_{\mathbf{x}\ge 0} \langle \mathbf{d}, \mathbf{x}\rangle \quad \text{s.t} \quad \mathbf{A}\mathbf{x} = \mathbf{b}. \tag{4}$$

with $\mathbf{d}^\top = (\text{vec}(\mathbf{C})^\top, \mathbf{0}_{2n}^\top), \mathbf{x}^\top = (\text{vec}(\mathbf{X})^\top, \mathbf{p}^\top, \mathbf{q}^\top), \mathbf{b}^\top = (\mathbf{r}^\top, \mathbf{c}^\top, s)$, and $\mathbf{A}$ is a constant matrix such that $(\mathbf{A}\mathbf{x})^\top = ((\mathbf{X}\mathbf{1}+\mathbf{p})^\top, (\mathbf{X}^\top\mathbf{1}+\mathbf{q})^\top, \mathbf{1}^\top\mathbf{X}\mathbf{1})$.

Applying Blondel's idea for a sparse transport plan [22], by adding a quadratic regularizer term, we derive the formulation for Quadratic POT (QPOT) as follows

$$\min_{\mathbf{x}\ge 0} f_\gamma(\mathbf{x}) := \langle \mathbf{d}, \mathbf{x}\rangle + \gamma\|\mathbf{x}\|^2 \,\text{s.t} \quad \mathbf{A}\mathbf{x} = \mathbf{b}. \tag{5}$$

Remark 2.1: In this work, we use APDAGD to solve QPOT, while CombiPOT is utilized to solve POT.

Remark 2.2: The algorithms proposed in this paper are built upon the extended Optimal Transport (OT) formulation introduced in [21]. However, when handling approximate solutions, this approach may introduce issues, which we will explicitly address later for each specific algorithm. In general, to resolve this challenge, we will employ the ROUND-POT rounding procedure as derived in [15] to obtain feasible solutions.

III. Combinatorial Algorithm for Approximating the POT

A. Preliminaries of Combinatorial OT (CombiOT)

Beyond the use of gradient methods and regularization techniques to leverage convex properties and improve convergence rates, OT has also been historically approached from a discrete optimization perspective, specifically as a minimum-cost flow problem, with an emphasis on graph-based algorithms ([23, 16]). Lahn et al [18] recently brought up an algorithm with competitive complexity $\mathscr{O}(n^2/\varepsilon^2)$ and also easy to parallelize. From this point, we refer to the GPU-friendly combinatorial algorithm introduced in [18] as CombiOT and we now introduce its main property for OT.

Theorem 3.1 (Theorem 2, [18]):

- Given a $\varepsilon > 0$, CombiOT computes a $\varepsilon-$approximate transport plan in $\mathscr{O}(n^2/\varepsilon^2)$ sequential time.
- In the Massive Parallel Computation (MPC) model, the algorithm executed in expected $\mathscr{O}(\log(n)/\varepsilon^2)$ parallel time or in $\mathscr{O}(\log(\log(n/\varepsilon))/\varepsilon^2)$ rounds with $\mathscr{O}(n/\varepsilon)$ memory per machine.

Remark 3.2: In order to adapt with the 3ε term in [18, Theorem 3], in this work, we set the error factor to $\varepsilon/3$.

Building on this algorithm and the previously discussed rounding procedure [15], we proceed to construct a POT solver that incorporates these advancements.

B. CombiPOT: An Extension of CombiOT in POT

We approximate the partial optimal transport solution using the extended optimal transport formulation from [21]. This formulation transforms the POT problem to an extended OT problem as:

$$\widetilde{\mathbf{C}} = \begin{pmatrix} \mathbf{C} & \mathbf{0}_n \\ \mathbf{0}_n^\top & A \end{pmatrix} \in \mathbb{R}_+^{(n+1)\times(n+1)}, \tag{6}$$

where $A > \max(C_{i,j})$, with the two marginals are augmented to $(n+1)$-dimensional vectors as $\widetilde{\mathbf{r}}^\top = (\mathbf{r}^\top, \|\mathbf{c}\|_1 - s)$ and $\widetilde{\mathbf{c}}^\top = (\mathbf{c}^\top, \|\mathbf{r}\|_1 - s)$.

If this extended problem admits a solution

$$\widetilde{\mathbf{X}} = \begin{pmatrix} \bar{\mathbf{X}} & \widetilde{\mathbf{p}} \\ \widetilde{\mathbf{q}}^\top & \widetilde{X}_{n+1,n+1} \end{pmatrix} \in \mathbb{R}_+^{(n+1)\times(n+1)},$$

then $\bar{\mathbf{X}}$ is the solution to the original POT problem ([21, Proposition 1]).

Algorithm 1 CombiPOT

Input: marginals $\mathbf{r}, \mathbf{c}$; cost matrix $\mathbf{C}$; dummy coefficient A_{mult}; transport mass s; accuracy ε

1: $\tilde{\varepsilon} = \frac{\varepsilon}{3}$
2: $\tilde{\mathbf{C}} \leftarrow \begin{pmatrix} \mathbf{C} & \mathbf{0}_n \\ \mathbf{0}_n^\top & A \end{pmatrix} \in \mathbb{R}_+^{(n+1)\times(n+1)}$
3: $\tilde{\mathbf{r}} \leftarrow \left[\mathbf{r}^\top, \|\mathbf{c}\|_1 - s\right]^\top$
4: $\tilde{\mathbf{c}} \leftarrow \left[\mathbf{c}^\top, \|\mathbf{r}\|_1 - s\right]^\top$
5: $\tilde{\mathbf{X}} \leftarrow \text{CombiOT}(\mathbf{C}, \tilde{\mathbf{r}}, \tilde{\mathbf{c}}, \tilde{\varepsilon})$
6: $\mathbf{X} \leftarrow \text{ROUND-POT}(\tilde{\mathbf{X}}, \tilde{\mathbf{r}}, \tilde{\mathbf{c}}, s)$

Output: $\mathbf{X}$

Algorithm 2 Approximating POT by APDAGD

Input: marginals $\mathbf{r},\mathbf{c}$; cost matrix $\mathbf{C}$; transport mass s

$\gamma = \frac{\varepsilon}{2s^2}, \tilde{\varepsilon} = \frac{\varepsilon}{8\|\mathbf{C}\|_{\max}}$
if $\|\mathbf{r}\|_1 > 1$ **then**
  $\tilde{\varepsilon} \leftarrow \min\left\{\tilde{\varepsilon}, 8\frac{\|\mathbf{r}\|_1 - s}{\|\mathbf{r}\|_1 - 1}\right\}$
end if
if $\|\mathbf{c}\|_1 > 1$ **then**
  $\tilde{\varepsilon} \leftarrow \min\left\{\tilde{\varepsilon}, 8\frac{\|\mathbf{c}\|_1 - s}{\|\mathbf{c}\|_1 - 1}\right\}$
end if
$\tilde{\mathbf{r}} \leftarrow \left(1 - \frac{\tilde{\varepsilon}}{8}\right)\mathbf{r} + \frac{\tilde{\varepsilon}}{8}\mathbf{1}_n$
$\tilde{\mathbf{c}} \leftarrow \left(1 - \frac{\tilde{\varepsilon}}{8}\right)\mathbf{c} + \frac{\tilde{\varepsilon}}{8}\mathbf{1}_n$
$\tilde{\mathbf{X}} \leftarrow \text{APDAGD}(\mathbf{C}, \gamma, \tilde{\mathbf{r}}, \tilde{\mathbf{c}}, \tilde{\varepsilon}/2)$
$\mathbf{X} \leftarrow \text{ROUND-POT}(\tilde{\mathbf{X}}, \tilde{\mathbf{r}}, \tilde{\mathbf{c}}, s)$

Output: $\mathbf{X}$

We first solve the extended OT problem using CombiOT, then pass it through ROUND-POT to obtain the solution for the POT problem. The algorithm is called CombiPOT (Algorithm 1. The following Theorem 3.3 illustrates CombiPOT's guarantee.

Theorem 3.3: Given an $\varepsilon > 0$, CombiPOT returns an $\varepsilon-$approximate solution in $\mathcal{O}(n^2/\varepsilon^2)$.

We provide the proof sketch for this theorem, the full proof can be found in the Appendix.

Proof sketch:

Step 1: We obtain the CombiOT $\varepsilon-$approximate solution $\widetilde{\mathbf{X}}$ of the extended OT problem and get the approximate solution $\overline{\mathbf{X}}$ for the POT problem by trimming the last row and the last column.

Step 2: We bound the constraint violation of $\overline{\mathbf{X}}$, which is equivalent to $\widetilde{X}_{n+1,n+1}$.[1]

Step 3: We combine the theoretical guarantees of CombiOT and ROUND-POT to establish that the complexity of CombiPOT is $\mathcal{O}(n^2/\varepsilon^2)$. ■

IV. Adaptive Primal-Dual Accelerated Gradient Descent (APDAGD)

We present the computational complexity of APDAGD for QPOT and its proof sketch. The detailed proof is included in the Appendix.

Theorem 4.1: Let $\varepsilon > 0$ denote the desired accuracy. Then the total computational complexity of using APDAGD to produce a $\varepsilon-$approximate plan is

$$\mathcal{O}\left(\frac{n^{2.5}}{\varepsilon}\right).$$

Proof sketch:

Step 1: Given the Lagrangian problem of (5) with dual variable $\lambda = (\mathbf{y},\mathbf{z},t)$ and $f_\gamma, \mathbf{A}, \mathbf{b}$ defined in (4) as follows

$$\max_{\lambda \in \mathbb{R}_+^{2n+1}} \left(\min_{\mathbf{x}} \left(f_\gamma(\mathbf{x}) - \langle \lambda, \mathbf{A}\mathbf{x} - \mathbf{b}\rangle\right)\right)$$

[1]This constraint violation is why we have to include Round-POT

We prove that this is equivalent to

$$\max_{\mathbf{y},\mathbf{z},t} \Big\{ g(\lambda) = \langle \mathbf{y}, \mathbf{r}\rangle + \langle \mathbf{z}, \mathbf{c}\rangle + ts - \sum_{i,j=1}^{n} \frac{\max(y_i + z_j + t - C_{i,j}, 0)^2}{4\gamma} - \sum_{i=1}^{n} \frac{\max(y_i, 0)^2}{4\gamma} - \sum_{i=1}^{n} \frac{\max(z_i, 0)^2}{4\gamma} \Big\}. \tag{7}$$

and the primal variables can be calculated via

$$X_{i,j} = \max\left\{0, \frac{y_i + z_j + t - C_{i,j}}{2\gamma}\right\}, \tag{8}$$

$$\mathbf{p}_i = \max\left\{0, \frac{y_i}{2\gamma}\right\}, \tag{9}$$

$$\mathbf{q}_j = \max\left\{0, \frac{z_j}{2\gamma}\right\}. \tag{10}$$

Remark 4.2: The presence of zeros in the primal results suggests a sparse transport plan.

Step 2: We establish the bound for the dual variable $\|\lambda\|_\infty = \mathcal{O}(\|\mathbf{C}\|_{\max})$ using the relations between primal and dual variables, which leads to $\|\lambda\|_2 = \mathcal{O}(\sqrt{n}\|\mathbf{C}\|_{\max})$. The explicit bound is provided in the following Lemma.

Lemma 1:

$$y_{inf} = -2\gamma\|\mathbf{r}\|_\infty - 2\gamma s - \|\mathbf{C}\|_\infty < y_i^* < 2\gamma c_i = y_{sup} \tag{11}$$

$$z_{inf} = -2\gamma\|\mathbf{c}\|_\infty - 2\gamma s - \|\mathbf{C}\|_\infty < z_i^* < 2\gamma r_i \tag{12}$$

$$t_{inf} = -2\gamma(\|\mathbf{c}\|_\infty + \|\mathbf{r}\|_\infty) < t^* \le 2\gamma s + \|\mathbf{C}\|_\infty = t_{sup} \tag{13}$$

This bound is essential in proving the APDAGD guarantees and, consequently, in determining the final complexity.

Step 3: We prove that $\gamma = 0.5\varepsilon/s^2$ is the best regularizing coefficient for speed, and with that γ, we need $\mathcal{O}(n^{0.5}/\varepsilon)$ iterations of APDAGD.

Step 4: Combining with the theoretical guarantees of APDAGD and ROUND-POT[2], we conclude the final computational complexity of APDAGD for QPOT to be $\mathcal{O}(n^{2.5}/\varepsilon)$. ■

The full algorithm is described followed as Algorithm 2.

V. Experiments

To further validate the quality of the proposed algorithms, we conduct various numerical studies to evaluate computational time, accuracy, transport plan sparsity, and total transport cost. Additionally, in all settings, the optimal cost is computed using Linear Programming via the `cvxpy` package with the GUROBI solver. Due to space constraints, we present only the experiments related to the Domain Adaptation task, while additional results can be found in the Appendix.

A. Domain Adaptation

In this section, we conduct experiments on the Domain Adaptation problem using various methods discussed in the paper. We divide the analysis into two parts: first, comparing regularized methods (QAPDAGD, APDAGD, and Sinkhorn),

[2]The APDAGD result plan involves a bounded constraint violation, which is why we need to apply ROUND-POT.

TABLE I: Standard hyperparameters for Domain Adaptation experiments

Hyperparameter	Value
Data points in source	300
Data points in target	400
Transport mass	$0.88 \times \min\{\|r\|_1, \|c\|_1\}$
Tolerance ε	$1e-3$
Noise level	0.05
Test sample size	10000

and second, comparing non-regularized methods (CombiPOT and Linear Programming). Our setup utilizes the "moon" dataset from scikit-learn with each dataset containing two crescent moons with the same amount of points. For the target dataset, it originates from the same distribution as the source, but the points undergo a 60-degree rotation (black scatter points), illustrating a domain covariate shift. We take a set of datapoints from each of the 2 moons. We use K-means clustering to map the higher-cardinality set to a set of K-means centers, matching the cardinality of the other set. The two marginals are built as the number of points contributing to each K-Means centroid in each set, then normalize to have sum of 1. The cost matrix is simply the squared euclidean distance between the centroids of each set. Then, we train a support vector machine (SVM) on the predicted target dataset and test it on the original source dataset. Additionally, we compute transport plan sparsity as the ratio of cells exceeding the sparsity threshold to the total number of cells, with the threshold defined per experiment. Unless otherwise stated in specific experiment setups, we set up our hyperparameters as listed in TABLE.I.

1) Comparision between regularized methods: In our first experiment, we aim to investigate the impact of test sample size on the performance of the solvers. To this end, we conduct experiments with test sample sizes ranging from 5,000 to 50,000 and set the transport mass to $s = 0.95 \times \min\{\|r\|_1, \|c\|_1\}$, Our results indicate that while Sinkhorn and APDAGD maintain accuracy levels of approximately 0.92 and 0.95, respectively, QAPDAGD consistently achieves an accuracy above 0.97 across all sample sizes. The values of each solver corresponding to a few test sample sizes are plotted in TABLE II.

In our second experiment, we investigate the impact of noise on the accuracy of each method. Keeping all other settings unchanged, we systematically increase the noise level in the test set from 0.01 to 0.4. The performance of all solvers under varying noise levels is presented in Fig.1-Left. Overall, as the noise level increases, the performance of all solvers

TABLE II: Comparison of accuracy across different solvers and sample sizes.

Test Sample Size	QAPDAGD	APDAGD	Sinkhorn
5000	**0.9721**	0.9480	0.9136
15000	**0.9729**	0.9482	0.9165
25000	**0.9724**	0.94704	0.91472
35000	**0.97196**	0.94783	0.9154
45000	**0.97226**	0.94758	0.91551

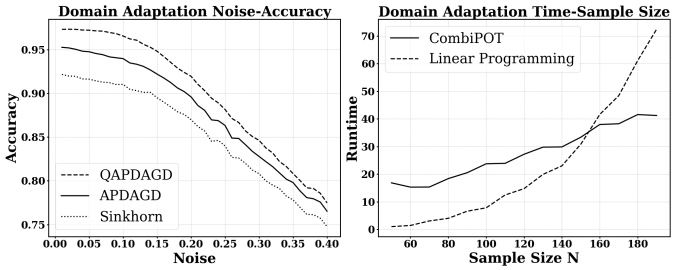


Fig. 1: Left: Accuracy comparison of QAPDAGD, APDAGD, and the Sinkhorn method across different noise levels ranging from 0.01 to 0.4. Right: Computational time comparison between the CombiPOT and Linear Programming methods for varying training sample sizes from 50 to 200.

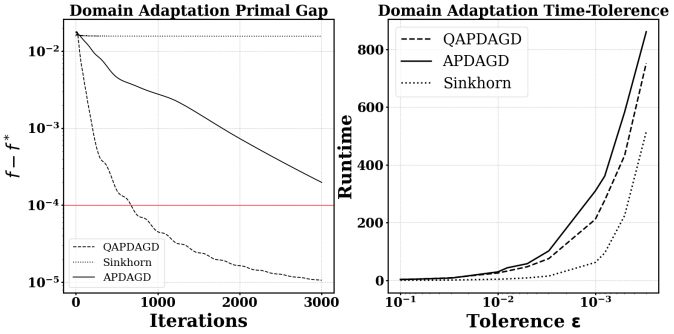
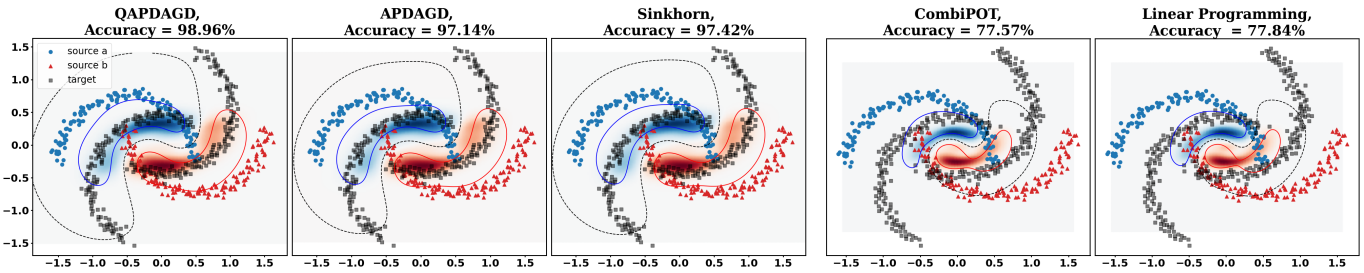


Fig. 2: Left: Primal Gap of each methods after each iteration. Right: Computational Time of QAPDAGD, APDAGD and Sinkhorn algorithm for different values of tolerance

declines. However, our proposed solver, QAPDAGD, consistently outperforms the other three across all noise levels. This demonstrates the effectiveness of our method in adaptability to noisy environments.

In the next experiment, we examine the performance of each method under different tolerance levels ε. Keeping all other settings unchanged, we vary ε from 10^{-1} to 3×10^{-4}. We evaluate runtime of the transportation matrix, plotting the values for each method at each tolerance level in Fig.2-Right. Additionally, we present the transport plan sparsity for each method, with a tolerance level ranging from 10^{-2} to 3×10^{-3} and a sparsity threshold of 10^{-6}, in Table III. Lastly, for accuracy evaluation, the SVM results of each methods corresponding to the tolerance $\varepsilon = 1e-3$ are shown in Fig. 3-(a). Overall, QAPDAGD outperforms the original APDAGD algorithm in terms of runtime, particularly for smaller tolerance levels. At low tolerance, QAPDAGD also achieves total costs comparable to the other two methods. Additionally, from the results shown, QAPDAGD demonstrates superior performance over APDAGD and Sinkhorn in terms of both sparsity and accuracy.

Lastly, we analyze the impact of training dataset size. Keeping all other settings unchanged, we set the number of data points in the source and target domains to n and $2n$, respectively, varying n from 350 to 800.The sparsity of each method, with a threshold of 10^{-6}, is shown in Table IV. Overall, QAPDAGD consistently achieves higher sparsity



(a) (b)

Fig. 3: (a): The SVM results of QAPDAGD, APDAGD, and Sinkhorn methods with a tolerance of 10^{-3}. (b): The SVM results for CombiPOT and Linear Programming methods with training sample size $n = 150$.

across all training sample sizes, demonstrating the scalability of our proposed method.

2) Comparision between non-regularized methods: In this experiment, we keep all settings unchanged while varying the sample size of the source and target distributions, setting them to n and $2n$, respectively, with n ranging from 50 to 200. The computation time for each method are recorded in Fig.1-Right. Additionally, we also present the SVM results for $n = 150$ obtained with both methods in Fig.3-(b). In general, we observe that CombiPOT outperforms Linear Programming in terms of runtime as the problem size increases, while still maintaining comparable accuracy. This highlights the significant scalability of CombiPOT. It should be noted that solving Linear Programming can be highly memory-intensive, with a worst-case space complexity of $O((x \times y)^{3.5} \log(1/\varepsilon))$, where x and y are the numbers of source and target data points, respectively.

B. Positive-Unlabeled Learning

An emerging application of POT is Positive-Unlabeled (PU) learning. PU classification is a variant of the traditional binary classification problem, also known as Positive-Negative (PN) learning, where instead of having access to both positive and negative labeled data, we only have access to a dataset containing positive and unlabeled samples.

1) Problem Settings: Given a dataset with feature vectors $X \in \mathbb{R}^d$ and labels $Y \in \{-1, 1\}$, we assume that the underlying joint distribution of (X, Y) is given by the density function $f(x, y)$. The class-conditional densities are defined as:

$$f_p(x) = P(X = x \mid Y = 1), \quad f_n(x) = P(X = x \mid Y = -1). \tag{14}$$

In a typical PN learning setting, we have access to a fully labeled dataset consisting of both positive and negative samples, and the goal is to learn a function $g : \mathbb{R}^d \to \{-1, 1\}$ that maps each data point to its corresponding label.

TABLE III: Comparison of sparsity across different solvers for varying tolerance levels.

Tolerance	QAPDAGD	APDAGD	Sinkhorn
1e-2	**0.9712**	0.9501	0.9665
8e-3	**0.9818**	0.9639	0.9705
5e-3	**0.9843**	0.9720	0.9776
3e-3	**0.9856**	0.9782	0.9834

TABLE IV: Comparison of Sparsity: QAPDAGD, APDAGD, and Sinkhorn Across Different Train Sample Sizes

Train Sample Size	QAPDAGD	APDAGD	Sinkhorn
350	**0.9943**	0.9822	0.9841
400	**0.9947**	0.9833	0.9852
450	**0.9954**	0.9837	0.9855
500	**0.9957**	0.9846	0.9861
550	**0.9960**	0.9853	0.9870
600	**0.9961**	0.9861	0.9876
650	**0.9965**	0.9864	0.9879
700	**0.9967**	0.9878	0.9888
750	**0.9967**	0.9878	0.9891
800	**0.9967**	0.9878	0.9891

In PU learning, however, we only have access to a dataset of positive samples and an unlabeled set, where unlabeled samples may belong to either the positive or negative class. Formally, let:

$$P = \{x_i\}_{i=1}^{n_p} \sim f_p(x), \quad U = \{x_j\}_{j=1}^{n_u} \sim f_u(x), \tag{15}$$

where $f_u(x)$ is the marginal distribution of the unlabeled data, defined as:

$$f_u(x) = \pi_p f_p(x) + (1 - \pi_p) f_n(x), \tag{16}$$

with $\pi_p = P(Y = 1)$ denoting the prior probability of the positive class.

The objective of PU learning is to estimate a classifier $g(x)$ that can separate positive and negative instances despite the absence of explicitly labeled negative samples.

2) Experimental Setup: To evaluate the performance of our approach, we benchmark five POT solvers: Linear Programming (LP), APDAGD, QAPDAGD, Sinkhorn, and QPDual. Following the setup in [21], we utilize a suite of benchmark datasets designed to test both the Selected Completely At Random (SCAR) and Selected At Random (SAR) labeling assumptions. All experiments are conducted with a 20% label noise level, and the POT algorithms are configured to transport only 80% of the total mass.

3) Accuracy Evaluation: Each solver is run using three randomly generated seeds per dataset. We report the mean accuracy (μ) and standard deviation (σ) across these runs in Table V.

Our results show that APDAGD and QAPDAGD consistently other POT solvers in 7 out of 8 datasets. Specifically, QAPDAGD achieves the highest accuracy on 5 out of 8

datasets, and APDAGD achieves the best accuracy on the remaining 3 datasets and shows lower variance in its predictions compared to Sinkhorn.

VI. CONCLUSIONS AND FUTURE WORKS

In this paper, we first propose a sparse solver to obtain a feasible solution for POT with quadratic regularization. We then introduce a combinatorial-based algorithm inspired by OT solvers, providing theoretical guarantees for the problem. While QAPDAGD achieves the best complexity in theory with a high level of sparsity in practice, CombiPOT demonstrates superior practical performance for reasonably sized problems. Despite these promising results, the combinatorial approach in this work is somewhat adapted rather than fully intrinsic. Developing a purely combinatorial POT solver—one that fully inherits the greedy nature of the original algorithm—is an ongoing effort. If successful, such an approach could offer even better complexity than existing methods for solving POT. Moreover, other potential avenues are to study and develop efficient computational methods for the Unbalanced Optimal Transport problem [24], stochastic optimization methods for these OT variants [25, 26] or develop decentralized optimization algorithms to study the distributed variant of the problem [27].

VII. SUPPLEMENTAL MATERIALS

For additional proofs and experiments, please refer to [link]

REFERENCES

[1] C. Villani, *Optimal transport: Old and New*. Springer, 2008.

[2] L. V. Kantorovich, "On the translocation of masses," in *Dokl. Akad. Nauk. USSR (NS)*, vol. 37, 1942, pp. 199–201.

[3] G. Fu, S. Liu, S. Osher, and W. Li, "High order computation of optimal transport, mean field planning, and potential mean field games," *Journal of Computational Physics*, vol. 491, p. 112346, Oct. 2023. [Online]. Available: http://dx.doi.org/10.1016/j.jcp.2023.112346

[4] G. Fu, S. Osher, W. Pazner, and W. Li, "Generalized optimal transport and mean field control problems for reaction-diffusion systems with high-order finite element computation," 2023. [Online]. Available: https://arxiv.org/abs/2306.06287

[5] Y. Chen, T. Georgiou, and M. Pavon, "On the relation between optimal transport and schrödinger bridges: A stochastic control viewpoint," 2014. [Online]. Available: https://arxiv.org/abs/1412.4430

[6] I. Haasler, A. Ringh, Y. Chen, and J. Karlsson, "Multimarginal optimal transport and schrödinger bridges on trees," *arXiv preprint arXiv:2004.06909*, 2020.

[7] M.-N. Phung and M.-B. Tran, "Control, optimal transport and neural differential equations in supervised learning," 2025. [Online]. Available: https://arxiv.org/abs/2503.15105

[8] Y. Balaji, R. Chellappa, and S. Feizi, "Robust optimal transport with applications in generative modeling and domain adaptation," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 934–12 944, 2020.

[9] K. Le, H. Nguyen, Q. M. Nguyen, T. Pham, H. Bui, and N. Ho, "On robust optimal transport: Computational complexity and barycenter computation," in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021. [Online]. Available: https://openreview.net/forum?id=xRLT28nnlFV

[10] A. Gramfort, G. Peyré, and M. Cuturi, "Fast optimal transport averaging of neuroimaging data," *CoRR*, vol. abs/1503.08596, 2015. [Online]. Available: http://arxiv.org/abs/1503.08596

[11] O. Pele and M. Werman, "A linear time histogram metric for improved sift matching," in *European Conference on Computer Vision*, 2008.

[12] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International journal of computer vision*, vol. 40, no. 2, pp. 99–121, 2000.

[13] J.-D. Benamou, G. Carlier, M. Cuturi, L. Nenna, and G. Peyré, "Iterative bregman projections for regularized transportation problems," *SIAM Journal on Scientific Computing*, vol. 37, no. 2, pp. A1111–A1138, 2015.

[14] K. Le, H. Nguyen, T. Pham, and N. Ho, "On multimarginal partial optimal transport: Equivalent forms and computational complexity," 2021. [Online]. Available: https://arxiv.org/abs/2108.07992

[15] A. D. Nguyen, T. D. Nguyen, Q. M. Nguyen, H. H. Nguyen, L. M. Nguyen, and K.-C. Toh, "On partial optimal transport: Revising the infeasibility of sinkhorn and efficient gradient methods," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 8, 2024, pp. 8090–8098.

[16] J. B. Orlin, "A polynomial time primal network simplex algorithm for minimum cost flows," *Mathematical Programming*, vol. 78, no. 2, pp. 109–129, Aug 1997. [Online]. Available: https://doi.org/10.1007/BF02614365

[17] N. Lahn, D. Mulchandani, and S. Raghvendra, "A graph theoretic additive approximation of optimal transport," *ArXiv Preprint: 1905.11830*, 2019.

[18] N. Lahn, S. Raghvendra, and K. Zhang, "A combinatorial algorithm for approximating the optimal transport in the parallel and mpc settings," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[19] P. A. Van den Elsen, E.-J. Pol, and M. A. Viergever, "Medical image matching-a review with classification," *IEEE Engineering in Medicine and Biology Magazine*, vol. 12, no. 1, pp. 26–39, 2002.

[20] L. Linsen, *Point cloud representation*. Univ., Fak. für Informatik, Bibliothek Technical Report, Faculty of Computer ..., 2001.

[21] L. Chapel, M. Z. Alaya, and G. Gasso, "Partial optimal tranport with applications on positive-unlabeled learning," in *Advances in Neural Information Processing Systems 33*, 2020.

TABLE V: Accuracy comparison of different solvers on eight PU Learning benchmark datasets.

	lp		apdagd		qapdagd		sinkhorn		qpdual	
dataset	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
connect-4	0.42	0.0	0.60	0.01	**0.62**	0.02	0.61	0.02	0.42	0.0
mushrooms	0.16	0.0	**0.92**	0.03	0.86	0.01	0.50	0.01	0.16	0.01
pageblocks	0.84	0.0	**0.90**	0.02	0.88	0.01	0.86	0.01	0.84	0.0
shuttle	0.65	0.0	**0.88**	0.0	0.85	0.01	0.82	0.05	0.65	0.0
spambase	0.11	0.0	**0.69**	0.02	0.68	0.02	0.68	0.13	0.11	0.0
usps	0.60	0.0	0.83	0.0	**0.92**	0.0	0.80	0.06	0.60	0.0
mnist	0.76	0.0	0.85	0.01	**0.87**	0.01	0.81	0.01	0.76	0.0
colored-mnist	0.76	0.0	0.76	0.0	**0.77**	0.0	**0.77**	0.02	0.75	0.0

Hyperparameter	**Value**
Regularization strength	0.005
Number of seeds	3
Transported mass	80%
Dataset noise level	20%

[22] M. Blondel, V. Seguy, and A. Rolet, "Smooth and sparse optimal transport," in *International conference on artificial intelligence and statistics*. PMLR, 2018, pp. 880–889.

[23] A. Phatak, S. Raghvendra, C. TRIPATHY, and K. Zhang, "Computing all optimal partial transports," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=gwcQajoXNF

[24] Q. M. Nguyen, H. H. Nguyen, Y. Zhou, and L. M. Nguyen, "On unbalanced optimal transport: Gradient methods, sparsity and approximation error," *Journal of Machine Learning Research*, vol. 24, no. 384, pp. 1–41, 2023. [Online]. Available: http://jmlr.org/papers/v24/22-1158.html

[25] H. H. Nguyen and S. T. Maguluri, "Stochastic approximation for nonlinear discrete stochastic control: Finite-sample bounds," 2023.

[26] L. Nguyen, P. H. NGUYEN, M. van Dijk, P. Richtarik, K. Scheinberg, and M. Takac, "SGD and hogwild! Convergence without the bounded gradients assumption," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 3750–3758. [Online]. Available: https://proceedings.mlr.press/v80/nguyen18c.html

[27] H. H. Nguyen, Y. Li, and T. Zhao, "Stochastic constrained decentralized optimization for machine learning with fewer data oracles: a gradient sliding approach," 2024. [Online]. Available: https://arxiv.org/abs/2404.02511

[28] P. Dvurechensky, A. Gasnikov, and A. Kroshnin, "Computational optimal transport: Complexity by accelerated gradient descent is better than by Sinkhorn's algorithm," in *International conference on machine learning*, 2018, pp. 1367–1376.

VIII. APPENDIX

IX. EXTRA ALGORITHMS

Algorithm 3 Adaptive Primal-Dual Accelerated Gradient Descent (APDAGD) [28, Algorithm 3]

Input: accuracy $\varepsilon_f, \varepsilon_{eq} > 0$; initial estimate L_0 such that $0 < L_0 < 2L$; initialize $M_{-1} = L_0$, $i_0 = k = \alpha_0 = \beta_0 = 0$, $\boldsymbol{\eta}_0 = \boldsymbol{\zeta}_0 = \boldsymbol{\lambda}_0 = \mathbf{0}$.

while $f(\hat{\mathbf{x}}_{k+1}) + \varphi(\boldsymbol{\eta}_{k+1}) \le \varepsilon_f, \|\mathbf{A}\hat{\mathbf{x}}_{k+1} - \mathbf{b}\|_2 \le \varepsilon_{eq}$ **do**
  while $\varphi(\boldsymbol{\eta}_{k+1}) \le \varphi(\boldsymbol{\lambda}_{k+1}) + \langle \nabla\varphi(\boldsymbol{\lambda}_{k+1}), \boldsymbol{\eta}_{k+1} - \boldsymbol{\lambda}_{k+1}\rangle + \frac{M_k}{2}\|\boldsymbol{\eta}_{k+1} - \boldsymbol{\lambda}_{k+1}\|_2^2$ **do**
    $M_k = 2^{i_k - 1} M_k$
    find α_{k+1} such that $\beta_{k+1} := \beta_k + \alpha_{k+1} = M_k \alpha_{k+1}^2$
    $\tau_k = \dfrac{\alpha_{k+1}}{\beta_{k+1}}$
    $\boldsymbol{\lambda}_{k+1} = \tau_k \boldsymbol{\zeta}_k + (1 - \tau_k)\boldsymbol{\eta}_k$
    $\boldsymbol{\zeta}_{k+1} = \boldsymbol{\zeta}_k - \alpha_{k+1}\nabla\varphi(\boldsymbol{\lambda}_{k+1})$
    $\boldsymbol{\eta}_{k+1} = \tau_k \boldsymbol{\zeta}_{k+1} + (1 - \tau_k)\boldsymbol{\eta}_k$
  end while
  $\hat{\mathbf{x}}_{k+1} = \tau_k \mathbf{x}(\boldsymbol{\lambda}_{k+1}) + (1 - \tau_k)\hat{\mathbf{x}}_k$
  $i_{k+1} = 0, k = k + 1$
end while

Output: $\hat{\mathbf{x}}_{k+1}, \boldsymbol{\eta}_{k+1}$.

Algorithm 4 ROUND-POT[15, Algorithm 1]

Input: $\mathbf{x} = (\mathrm{vec}(\mathbf{X})^\top, \mathbf{p}^\top, \mathbf{q}^\top)^\top$; marginals $\mathbf{r}$, $\mathbf{c}$; mass s.

$\bar{\mathbf{p}} = \mathrm{EP}(\mathbf{r}, s, \mathbf{p})$
$\bar{\mathbf{q}} = \mathrm{EP}(\mathbf{c}, s, \mathbf{q})$
$\mathbf{g} = \min\{\mathbf{1}, (\mathbf{r} - \bar{\mathbf{p}}) \oslash \mathbf{X}\mathbf{1}\}$
$\mathbf{h} = \min\{\mathbf{1}, (\mathbf{c} - \bar{\mathbf{q}}) \oslash \mathbf{X}^\top\mathbf{1}\}$
$\mathbf{X}' = \mathrm{diag}(\mathbf{g})\mathbf{X}\,\mathrm{diag}(\mathbf{h})$
$\mathbf{e}_1 = (\mathbf{r} - \bar{\mathbf{p}}) - \mathbf{X}'\mathbf{1}, \mathbf{e}_2 = (\mathbf{c} - \bar{\mathbf{q}}) - \mathbf{X}'^\top\mathbf{1}$
$\bar{\mathbf{X}} = \mathbf{X}' + \mathbf{e}_1\mathbf{e}_2^\top / \|\mathbf{e}_1\|_1$

Output: $\bar{\mathbf{x}} = (\bar{\mathbf{X}}, \bar{\mathbf{p}}, \bar{\mathbf{q}})$

Algorithm 5 CombiOT

[18]

Input: Demands SA, Sources DB; Cost $\mathbf{C}$; accuracy ε.

$\mathbf{C} \leftarrow \varepsilon \lfloor \mathbf{C}/\varepsilon \rfloor$
$M \leftarrow \emptyset$
Dual weights $y(a) = 0 \quad \forall a \in \mathbb{A}, y(b) = \varepsilon \quad \forall b \in B$
$B' \leftarrow$ free vertices of B
while $\|B'\| > \varepsilon n$ **do**
  $E' \leftarrow$ set of admissible edges with at least one end point in B'.
  $A' = \{a | a \in A \text{ and } (a,b) \in E'\}$
  Compute maximal matching M' in $G'(A' \cup B', E')$
  $A'' \leftarrow a$s matched in both M and M'
  $M'' \leftarrow$ edges of M incident on some vertex of A''.
  Adds edges of M' and deletes edges of M'' in M.
  $y(a) \leftarrow y(a) - \varepsilon \quad \forall (a,b) \in M'$
  $y(b) \leftarrow y(b) + \varepsilon \quad \forall b \in B'$ free with respect to M'
end while
Arbitrary match free vertices and return resulting matching

Output: Transport plan F, dual values yA, yB, totalcost, iteration

X. Proofs of Theorems and Lemma

A. Proof of Theorem 3.3

Proof:

From step 5 of Algorithm 1, we obtain a CombiOT $\varepsilon-$approximate solution (since $\varepsilon = 3\widetilde{\varepsilon}$) for the extended OT problem:

$$\widetilde{\mathbf{X}} = \begin{pmatrix} \bar{\mathbf{X}} & \widetilde{\mathbf{p}} \\ \widetilde{\mathbf{q}}^\top & \widetilde{X}_{n+1,n+1} \end{pmatrix} \in \mathbb{R}_+^{(n+1)\times(n+1)},$$

We also denote the optimal solution of the extended OT problem [21] as:

$$\widetilde{\mathbf{X}}_0 = \begin{pmatrix} \bar{\mathbf{X}}_0 & \widetilde{\mathbf{p}}_0 \\ \widetilde{\mathbf{q}}_0^\top & 0 \end{pmatrix} \in \mathbb{R}_+^{(n+1)\times(n+1)},$$

where $\bar{\mathbf{X}}_0 = \hat{\mathbf{X}}$ as per [21, Proposition 1] and the entry at position $(n+1, n+1)$ is zero ($A > \|\mathbf{C}\|_\infty$)[21, Appendix A2.2.3].

First we calculate the difference in cost of the extended POT solutions using the notations in (6):

$$\begin{aligned}\langle \widetilde{\mathbf{C}}, \widetilde{\mathbf{X}} - \widetilde{\mathbf{X}}_0 \rangle &= \left\langle \begin{pmatrix} \mathbf{C} & \mathbf{0}_n \\ \mathbf{0}_n^\top & A \end{pmatrix}, \begin{pmatrix} \bar{\mathbf{X}} - \bar{\mathbf{X}}_0 & \widetilde{\mathbf{p}} - \widetilde{\mathbf{p}}_0 \\ \widetilde{\mathbf{q}}^\top - \widetilde{\mathbf{q}}_0^\top & \widetilde{X}_{n+1,n+1} \end{pmatrix} \right\rangle \\ &= \langle \mathbf{C}, \bar{\mathbf{X}} - \hat{\mathbf{X}} \rangle + A\widetilde{X}_{n+1,n+1}\end{aligned}$$

According to Theorem 3.1, $\widetilde{\mathbf{X}}$ is an $\varepsilon-$approximate solution of the extended OT problem, therefore

$$\langle \mathbf{C}, \bar{\mathbf{X}} - \hat{\mathbf{X}} \rangle + A\widetilde{X}_{n+1,n+1} \leq \varepsilon$$

We now derive an upper bound for the cost difference $\langle \mathbf{C}, \mathbf{X} \rangle - \langle \mathbf{C}, \hat{\mathbf{X}} \rangle$ using the previous bound and Holder's inequality

$$\begin{aligned}\langle \mathbf{C}, \mathbf{X} \rangle - \langle \mathbf{C}, \hat{\mathbf{X}} \rangle &= (\langle \mathbf{C}, \mathbf{X} \rangle - \langle \mathbf{C}, \bar{\mathbf{X}} \rangle) + (\langle \mathbf{C}, \bar{\mathbf{X}} \rangle - \langle \mathbf{C}, \hat{\mathbf{X}} \rangle) \\ &\leq \langle \mathbf{C}, \mathbf{X} - \bar{\mathbf{X}} \rangle + (\varepsilon - A\widetilde{X}_{n+1,n+1}) \\ &\leq \|\mathbf{C}\|_\infty \left\|\mathbf{X} - \bar{\mathbf{X}}\right\|_1 + (\varepsilon - A\widetilde{X}_{n+1,n+1}) \\ &\leq \|\mathbf{C}\|_\infty \|\mathbf{x} - \widetilde{\mathbf{x}}\|_1 + (\varepsilon - A\widetilde{X}_{n+1,n+1}) \qquad (17)\end{aligned}$$

The last inequality is due to the fact that $(\mathbf{x} - \bar{\mathbf{x}})^\top = (\text{vec}(\mathbf{X} - \bar{\mathbf{X}})^\top, (\mathbf{p} - \widetilde{\mathbf{p}})^\top, (\mathbf{q} - \widetilde{\mathbf{q}})^\top)$

Next, we bound $\|\mathbf{x} - \widetilde{\mathbf{x}}\|_1$ by using the guarantee of ROUND-POT from [15, Theorem 6] to get

$$\|\mathbf{x} - \widetilde{\mathbf{x}}\|_1 \leq 23 \|\mathbf{A}\widetilde{\mathbf{x}} - \mathbf{b}\|_1 \qquad (18)$$

with $\mathbf{A}, \mathbf{b}$ defined as in (4).

Next, we analyze the term on the right hand side of the previous inequality as follow

$$\|\mathbf{A}\widetilde{\mathbf{x}} - \mathbf{b}\|_1 = \left\|\bar{\mathbf{X}}\mathbf{1} + \widetilde{\mathbf{p}} - \mathbf{r}\right\|_1 + \left\|\bar{\mathbf{X}}^\top \mathbf{1} + \widetilde{\mathbf{q}} - \mathbf{c}\right\|_1 + |\mathbf{1}^\top \bar{\mathbf{X}} \mathbf{1} - s|$$

Since $\widetilde{\mathbf{X}} \in \mathscr{U}(\widetilde{\mathbf{r}}, \widetilde{\mathbf{c}}, \widetilde{s} = \|\mathbf{r}\|_1 + \|\mathbf{c}\|_1 - s)$, we have

$$\begin{aligned}&\bar{\mathbf{X}}\mathbf{1} + \widetilde{\mathbf{p}} = \mathbf{r}, \quad \bar{\mathbf{X}}^\top \mathbf{1} + \widetilde{\mathbf{q}} = \mathbf{c} \text{ and} \\ &\mathbf{1}^\top \bar{\mathbf{X}} \mathbf{1} + \|\widetilde{\mathbf{p}}\|_1 + \|\widetilde{\mathbf{q}}\|_1 + \widetilde{X}_{n+1,n+1} = \|\mathbf{r}\|_1 + \|\mathbf{c}\|_1 - s.\end{aligned}$$

These conditions lead to

$$\begin{aligned}\mathbf{1}^\top \bar{\mathbf{X}} \mathbf{1} + \|\widetilde{\mathbf{p}}\|_1 &= \|\widetilde{\mathbf{r}}\|_1, \\ \mathbf{1}^\top \bar{\mathbf{X}} \mathbf{1} + \|\widetilde{\mathbf{q}}\|_1 &= \|\widetilde{\mathbf{c}}\|_1, \\ \mathbf{1}^\top \bar{\mathbf{X}} \mathbf{1} + \|\widetilde{\mathbf{p}}\|_1 + \|\widetilde{\mathbf{q}}\|_1 + \widetilde{X}_{n+1,n+1} &= \|\mathbf{r}\|_1 + \|\mathbf{c}\|_1 - s.\end{aligned}$$

Adding the first two identities then subtract by the final one, we get $\mathbf{1}^\top \bar{\mathbf{X}} \mathbf{1} - s = \widetilde{X}_{n+1,n+1}$.

Therefore, $\|\mathbf{A}\widetilde{\mathbf{x}} - \mathbf{b}\|_1 = \widetilde{X}_{n+1,n+1}$ and by combining with (18), we have $\|\mathbf{x} - \widetilde{\mathbf{x}}\| \leq 23\widetilde{X}_{n+1,n+1}$

Substituting this result into (17), we get

$$\langle \mathbf{C}, \mathbf{X} \rangle - \langle \mathbf{C}, \hat{\mathbf{X}} \rangle \leq 23 \|\mathbf{C}\|_\infty \widetilde{X}_{n+1,n+1} + (\varepsilon - A\widetilde{X}_{n+1,n+1}).$$

By setting A to $23\|\mathbf{C}\|_\infty$, we conclude that CombiPOT produces $\mathbf{X}$, an $\varepsilon-$approximate solution for the POT problem.

Since the algorithm takes 1 round of CombiOT with complexity $\mathscr{O}(n^2/\varepsilon^2)$ and 1 round of ROUND-POT, the total complexity of CombiPOT is $\mathscr{O}(n^2/\varepsilon^2)$.

Remark 10.1: Here, we emphasize that since $\widetilde{X}_{n+1,n+1}$ might not be zero due to $\widetilde{\mathbf{X}}$ being an approximate solution rather than an optimal one, $\overline{\mathbf{X}}$ exhibits a constraint violation, highlighting the need to apply ROUND-POT.

■

B. Lagrangian duality

Consider an optimization problem as follows:

$$\min \quad f_0(x) \tag{19}$$
$$\text{s.t} \quad f_i(x) \le 0 \quad \forall i = 1,...,n \tag{20}$$

Let p^* be the solution to (19). We further denote the Lagrangian of (19) to be:

$$L(\lambda, x) = f_0(x) + \sum_{i=1}^{n} \lambda_i f_i(x)$$

with $\lambda_i \ge 0 \quad \forall i \in \{1,...,n\}$

Now, by setting $g(\lambda) = \min_x L(\lambda, x)$, we can formulate our problem into a dual version. Our target would be to find d^* satisfying:

$$d^* = \max_{\lambda} g(\lambda) \tag{21}$$

$g(\lambda)$ and (21) are known as the Lagrange dual function and the Lagrangian dual problem . Additionally, the solution d^* of (21) is called the dual solution and is always less than or equal to p^*, the primal solution . Strong duality holds when $p^* = d^*$. This occurs only if the optimization problem is convex and a strictly feasible point exists, i.e., a point x that strictly satisfies all constraints.

C. Lagrangian Dual of Quadratic Optimal Transport

Let $\mathbf{y} \in \mathbb{R}^n, \mathbf{z} \in \mathbb{R}^n$ and $t \in \mathbb{R}$ be the dual variables corresponding to the three equality constraints. The Lagrangian is

$$\begin{aligned}\mathscr{L}(\mathbf{x}, \gamma) &= \langle \mathbf{C}, \mathbf{X} \rangle + \gamma(\|\mathbf{X}\|^2 + \|\mathbf{p}\|^2 + \|\mathbf{q}\|^2) \\ &\quad - \langle \mathbf{y}, \mathbf{X}\mathbf{1} + \mathbf{p} - \mathbf{r} \rangle - \langle \mathbf{z}, \mathbf{X}^\top \mathbf{1} + \mathbf{q} - \mathbf{c} \rangle - t(\mathbf{1}^\top \mathbf{X} \mathbf{1} - s) \\ &= \langle \mathbf{y}, \mathbf{r} \rangle + \langle \mathbf{z}, \mathbf{c} \rangle + st + \langle \mathbf{C}, \mathbf{X} \rangle + \gamma \|\mathbf{X}\|^2 \langle \mathbf{y}, \mathbf{X}\mathbf{1} \rangle - \langle \mathbf{z}, \mathbf{X}^\top \mathbf{1} \rangle \\ &\quad - t\mathbf{1}^\top \mathbf{X} \mathbf{1} + \gamma \|\mathbf{p}\|^2 - \langle \mathbf{y}, \mathbf{p} \rangle + \gamma \|\mathbf{q}\|^2 - \langle \mathbf{z}, \mathbf{q} \rangle.\end{aligned}$$

So, we have the dual problem:

$$\begin{aligned}&\max_{\mathbf{y},\mathbf{z},t} F_\gamma(\mathbf{x} = (\mathbf{y}, \mathbf{z}, t)) \\ &= \left\{ \langle \mathbf{y}, \mathbf{r} \rangle + \langle \mathbf{z}, \mathbf{c} \rangle + ts + \min_{X_{i,j} \ge 0} \sum_{i,j=1}^{n} \gamma X_{i,j}^2 + X_{i,j}(C_{i,j} - y_i - z_j - t) \right. \\ &\quad \left. + \min_{p_i \ge 0} \sum_{i=1}^{n} p_i(\gamma p_i - y_i) + \min_{q_j \ge 0} \sum_{j=1}^{n} q_j(\gamma q_j - z_j) \right\}.\end{aligned} \tag{22}$$

Next, we minimize the Lagrangian w.r.t. $\mathbf{X}$, $\mathbf{p}$ and $\mathbf{q}$ according to the first order condition, which gives

$$X_{i,j} = -\min\left\{0, \frac{C_{i,j} - y_i - z_j - t}{2\gamma}\right\} = \max\left\{0, \frac{y_i + z_j + t - C_{i,j}}{2\gamma}\right\}, \tag{23}$$

$$\mathbf{p}_i = \max\left\{0, \frac{y_i}{2\gamma}\right\}, \tag{24}$$

$$\mathbf{q}_j = \max\left\{0, \frac{z_j}{2\gamma}\right\}. \tag{25}$$

Proof:

The optimal value for $X_{i,j} \ge 0$ is depending only on $X_{i,j}^2 + X_{i,j}(C_{i,j} - y_i - z_j - t)$, a quadratic function.

If $-\dfrac{C_{i,j} - y_i - z_j - t}{2\gamma} \ge 0$, we can choose $X_{i,j}$ to be that value. But else, the function increases on $\mathbb{R}_{\ge 0}$, so $X_{i,j} = 0$ is optimal. Same arguments apply for finding optimal p_i, q_i since they also only depend on quadratic functions. ■

Plugging this back to the dual problem (22), we have

$$\begin{aligned}\max_{\mathbf{y},\mathbf{z},t} \Bigg\{ & g(\mathbf{x} = (\mathbf{y}, \mathbf{z}, t)) = \langle \mathbf{y}, \mathbf{r} \rangle + \langle \mathbf{z}, \mathbf{c} \rangle + ts \\ & - \sum_{i,j=1}^{n} \frac{\max(y_i + z_j + t - C_{i,j}, 0)^2}{4\gamma} \\ & - \sum_{i=1}^{n} \frac{\max(y_i, 0)^2}{4\gamma} - \sum_{i=1}^{n} \frac{\max(z_i, 0)^2}{4\gamma} \Bigg\}.\end{aligned} \tag{26}$$

D. Proof of Lemma 1

We first find upper bounds for y_i^* and z_j^*, to derive the lower bound and upper bound for t^*. Finally, we use upper bound for t^* to derive lower bound for y_i^*, z_j^*.

For all i such that y_i^* is positive

$$y_i^* = 2\gamma p_i \le 2\gamma c_i$$

So in general, $y_i^* \le \max\{0, 2\gamma c_i\} \le 2\gamma c_i$.

Similarly, we also have $z_j^* \le 2\gamma \|\mathbf{r}\|_\infty$.

Since $\mathbf{X}_\gamma$ has positive s as sum of all of its nonnegative entries, it is not a zero matrix. So there exist i, j such that $X_{i,j,\gamma} > 0$, and of course $X_{i,j,\gamma} \le s$, then for that i, j we have

$$\begin{aligned}&0 < -C_{ij} + y_i^* + z_j^* + t^* \le 2\gamma s \\ \Rightarrow & t^* > C_{ij} - y_i^* - z_j^* > -2\gamma(\|\mathbf{c}\|_\infty + \|\mathbf{r}\|_\infty) \\ & \text{(based on the bounds for } y_i \text{ and } z_j \text{ found above)}\end{aligned}$$

Since $\|\mathbf{X}\|_1 = s < \min(\|\mathbf{r}\|_1, \|\mathbf{c}\|_1), \mathbf{p}^*, \mathbf{q}^*$ are not zero vectors, which means there exist i, j such that p_i^*, q_j^* are positive numbers, and therefore, from (9) and (10), y_i, z_j are positive

numbers too.
So we have

$$X_{i,j} = \max\left\{0, \frac{y_i^* + z_j^* + t^* - C_{i,j}}{2\gamma}\right\} \le s,$$
$$\Rightarrow y_i^* + z_j^* + t^* - C_{ij} \le \max(0, 2\gamma s) \le 2\gamma s$$
$$\Rightarrow t^* \le 2\gamma s + C_{ij} - y_i^* - z_j^* \le 2\gamma s + \|\mathbf{C}\|_\infty$$

We conclude the proof with lower bounds for y_i^* and z_j^*

$$\begin{aligned} & \forall i, \text{if } y_i^* < 0, p_i^* = 0 \\ \Rightarrow \quad & \mathbf{X}\mathbf{1}_i = c_i = r_i > 0 \\ \Rightarrow \quad & \exists j, X_{ij} > 0 \Rightarrow y_i^* + z_j^* + t^* - C_{ij} > 0 \\ \Rightarrow \quad & y_i^* > C_{ij} - z_j^* - t^* \ge -2\gamma\|\mathbf{r}\|_\infty - 2\gamma s - \|\mathbf{C}\|_\infty. \end{aligned}$$

So,

$$\begin{aligned} y_i^* &> \min\{0, -2\gamma\|\mathbf{r}\|_\infty - 2\gamma s - \|\mathbf{C}\|_\infty\} \\ &\ge -2\gamma\|\mathbf{r}\|_\infty - 2\gamma s - \|\mathbf{C}\|_\infty. \end{aligned}$$

And similarly, $z_j^* > -2\gamma\|\mathbf{c}\|_\infty - 2\gamma s - \|\mathbf{C}\|_\infty$.
With $\gamma = 0.5\varepsilon/s^2$, terms involving γ are infinitesimal compared to $\|\mathbf{C}\|_\infty$. Therefore $\|\lambda\| = \|(\mathbf{y}, \mathbf{z}, t)\| = \mathcal{O}(\|\mathbf{C}\|_\infty)$.

E. Proof of Theorem 4.1

Let $\mathbf{x}^{\text{POT}} = (\mathbf{X}^{\text{POT}}, \mathbf{p}^{\text{POT}}, \mathbf{q}^{\text{POT}})$ be a solution of 5 and $\overline{\mathbf{x}} = \text{ROUND-POT}(\hat{\mathbf{x}}_k)$, we have

$$\begin{aligned} \langle \mathbf{d}, \mathbf{x}^{\text{POT}} \rangle + \gamma \left\| \mathbf{x}^{\text{POT}} \right\|^2 &\ge \langle \mathbf{d}, \mathbf{x}_f \rangle + \gamma \left\| \mathbf{x}_f \right\|^2 \\ &\ge \langle \mathbf{d}, \hat{\mathbf{x}}_k \rangle + \gamma \|\hat{\mathbf{x}}_k\|^2 - \frac{32R^2}{\gamma k^2} \\ \text{So }, \langle \mathbf{d}, \hat{\mathbf{x}}_k - \mathbf{x}^{\text{POT}} \rangle &\le \gamma \left(\left\| \mathbf{x}^{\text{POT}} \right\|^2 - \|\hat{\mathbf{x}}_k\|^2 \right) + \frac{32R^2}{\gamma k^2} \\ &\le \gamma s^2 + \frac{32R^2}{\gamma k^2} \end{aligned}$$

From [15], Theorem 6, we have that:

$$\begin{aligned} \|\hat{\mathbf{x}}_k - \overline{\mathbf{x}}\|_1 &\le 23 \|\mathbf{A}\hat{\mathbf{x}}_k - \mathbf{b}\|_1 \\ &\le 23\sqrt{2n+1} \|\mathbf{A}\hat{\mathbf{x}}_k - \mathbf{b}\|_2 \\ \langle \mathbf{d}, \overline{\mathbf{x}} - \hat{\mathbf{x}}_k \rangle &\le \|\mathbf{d}\|_\infty \|\hat{\mathbf{x}}_k - \overline{\mathbf{x}}\|_1 \le \|\mathbf{C}\|_\infty \cdot 23\sqrt{2n+1} \|\mathbf{A}\hat{\mathbf{x}}_k - \mathbf{b}\|_2 \\ &= \|\mathbf{C}\|_\infty \cdot 23\sqrt{2n+1} \frac{32R}{\gamma k^2} \end{aligned}$$

Adding them up, we have:

$$\langle \mathbf{C}, \overline{\mathbf{X}} - \mathbf{X}^{\text{POT}} \rangle = \langle \mathbf{d}, \overline{\mathbf{x}} - \mathbf{x}^{\text{POT}} \rangle \le \gamma s^2 + \frac{32R^2}{\gamma k^2} \left(1 + \frac{23 \|\mathbf{C}\|_\infty \sqrt{2n+1}}{R} \right) \quad (27)$$

Using AM-GM, we know the best γ to minimize the right-hand side to ε would satisfy $\gamma s^2 = 0.5\varepsilon$, or $\gamma = \varepsilon/2s^2$.
The other component of the sum must also be 0.5ε. Therefore

$$\begin{aligned} 0.5\varepsilon &= \frac{32R^2}{\gamma k^2} \left(1 + \frac{23 \|\mathbf{C}\|_\infty \sqrt{2n+1}}{R} \right) \\ &= \frac{64(sR)^2}{\varepsilon k^2} \left(1 + \frac{23 \|\mathbf{C}\|_\infty \sqrt{2n+1}}{R} \right) \\ \Rightarrow k^2 &= \frac{128(sR)^2}{\varepsilon^2} \left(1 + \frac{23 \|\mathbf{C}\|_\infty \sqrt{2n+1}}{R} \right) \end{aligned}$$

Since $R = \mathcal{O}(\|\mathbf{C}\|_\infty \sqrt{n})$, the bracket is $\mathcal{O}(1)$. So the optimal number of loops is $k = \mathcal{O}(n^{\frac{1}{2}} \|\mathbf{C}\|_\infty / \varepsilon)$, combining with $\mathcal{O}(n^2)$ for each iteration of APDAGD gives the total complexity of $\mathcal{O}(n^{\frac{5}{2}} \|\mathbf{C}\|_\infty / \varepsilon)$.

F. Color Transfer

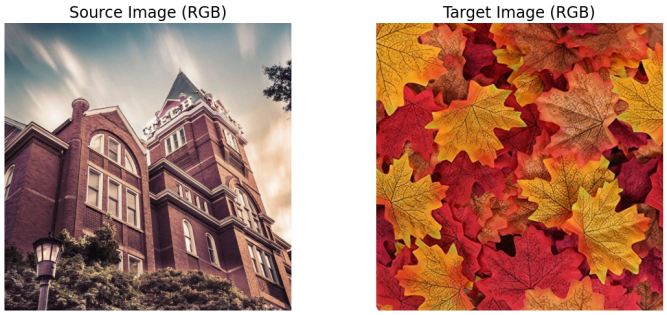


Fig. 4: Dataset for Color Transfer

Color transfer is an image editing technique that aligns the color palette of one image with another by transferring hue, saturation, and brightness while preserving contextual details. An RGB image is defined as $\mathbf{x} \in \mathbb{R}^{h \times w \times 3}$, where each pixel at coordinate (i, j) is represented as $x_{ij} = \{\text{red, green, blue}\}$ with values ranging from 0 to 255. This forms a color distribution across channels, and color transfer aims to match the source image's distribution to the target's. Our dataset consists of a pair of 256×256 images (Figure 4) with distinct color palettes. Instead of working in RGB, we convert images to LUV, where color is encoded in the U and V channels, reducing dimensionality from three (RGB) to two (UV) while better reflecting perceptual color differences. The conversion follows: $L = R + G + B$, $U = G/L$, and $V = B/L$. We extract color histograms with n bins by computing U and V frequencies, smoothing, and normalizing them to ensure $\|\mathbf{r}\|_1 = 1$ and $\|\mathbf{c}\|_1 = 1$. The cost matrix is defined as $C_{i,j} = \|a_i - b_j\|_2^2$, where a_i and b_j are histogram bin values.
The first experiment examines the transport mass, defined as $\alpha \times \min(\|r\|_1, \|c\|_1)$. Here, we vary α within the interval

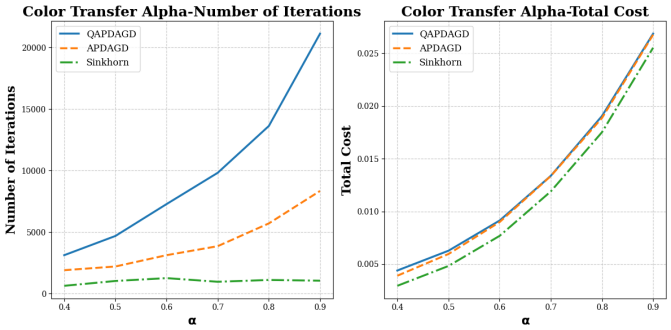


Fig. 5: Number of iterations and total cost for each value of alpha

$[0.4, 0.9]$ and evaluate the number of iterations, and total cost for QAPDAGD, APDAGD, and Sinkhorn at each value of α in Figure 5. Additionally, we have also plotted out the sparsity of each method when $\alpha = 0.8$ in Figure 6. Overall, although QAPDAGD requires more iterations, its total cost remains comparable to other methods. Moreover, it achieves slightly better sparsity in the transportation plan compared to APDAGD and Sinkhorn.

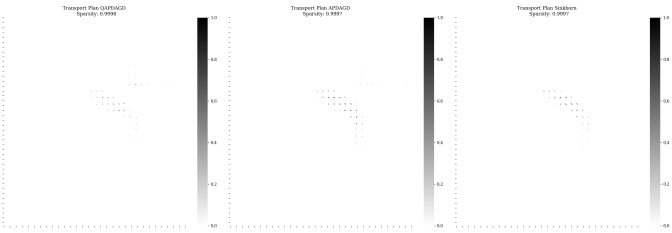


Fig. 6: Sparsity, number of iterations and total cost for each value of alpha

In the second experiment, we set $s = 0.8 \times \min(\|\mathbf{r}\|_1, \|\mathbf{c}\|_1)$ and tolerance $\varepsilon = 10^{-2}$, varying the sample size, i.e., the dimensions of the input and output images. Specifically, we set the image size to $n^2 \times n^2$ for n ranging from 20 to 40. Figure 8 shows the total cost and computation time for each method across different sample sizes. Additionally, for $n = 25$, Figure 9 presents the heatmaps of the transportation plan for each method. In general, the total costs and sparsity of all methods are approximately the same. However, due to its inherent structure, QAPDAGD exhibits a higher runtime compared to other methods as the sample size increases.

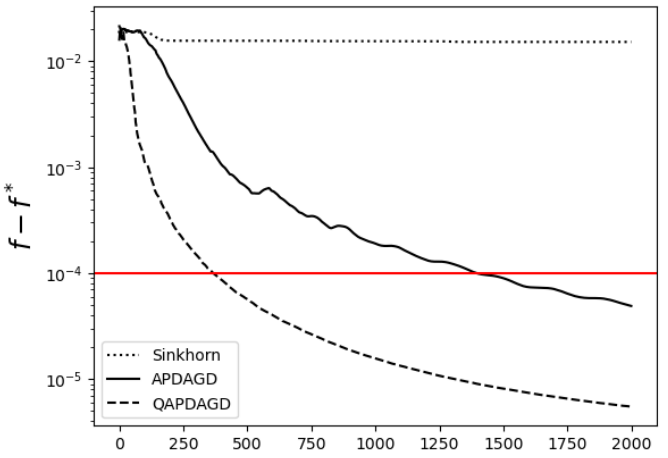


Fig. 7: Primal gap achieved by our QAPDAGD algorithm versus APDAGD and revised Sinkhorn for POT with same number of iterations and tolerance ε in red. The result transport plans are later used for Color Transfer.

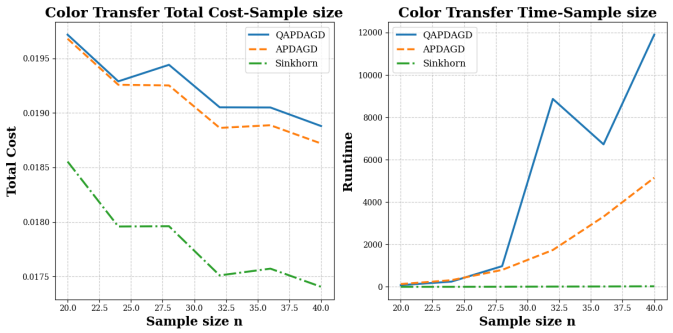


Fig. 8: Number of iterations and time taken for each value of sample size n.

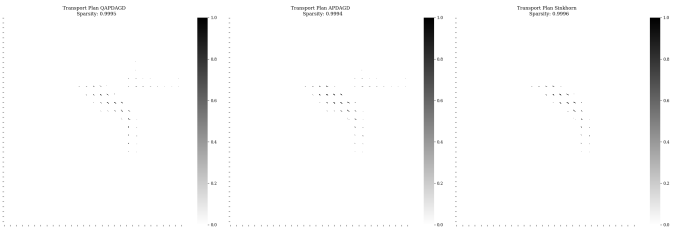


Fig. 9: The heatmaps of the transportation plans of each methods with the sample size $n = 25$.