Fast and Accurate Approximations of the Optimal Transport in Semi-Discrete and Discrete Settings

Pankaj K. Agarwal*   Sharath Raghvendra†   Pouyan Shirzadian‡

Keegan Yao*

Abstract

Given a d-dimensional continuous (resp. discrete) probability distribution μ and a discrete distribution ν, the semi-discrete (resp. discrete) Optimal Transport (OT) problem asks for computing a minimum-cost plan to transport mass from μ to ν; we assume n to be the number of points in the support of the discrete distributions. In this paper, we present three approximation algorithms for the OT problem with strong theoretical guarantees.

(i) *Additive approximation for semi-discrete OT:* For any parameter $\varepsilon > 0$, we present an algorithm that computes a semi-discrete transport plan τ with cost $\mathdollar(\tau) \leq \mathdollar(\tau^*) + \varepsilon$ in $n^{O(d)} \log \frac{\Delta}{\varepsilon}$ time; here, τ^* is the optimal transport plan, Δ is the diameter of the supports of μ and ν, and we assume we have access to an oracle that outputs the mass of μ inside a constant-complexity region in $O(1)$ time. Our algorithm works for several ground distances including the L_p-norm and the squared-Euclidean distance.

(ii) *Relative approximation for semi-discrete OT:* For any parameter $\varepsilon > 0$, we present an algorithm that computes a semi-discrete transport plan τ with cost $\mathdollar(\tau) \leq (1 + \varepsilon)\mathdollar(\tau^*)$ in $n\varepsilon^{-O(d)} \log(n) \log^{O(d)}(\log n)$ time; here, τ^* is the optimal transport plan, and we assume we have access to an oracle that outputs the mass of μ inside an orthogonal box in $O(1)$ time, and the ground distance is any L_p norm.

(iii) *Relative approximation for discrete OT:* For any parameter $\varepsilon > 0$, we present a Monte-Carlo algorithm that computes a transport plan σ with an expected cost $\mathdollar(\sigma) \leq (1 + \varepsilon)\mathdollar(\sigma^*)$ under any L_p norm in $n\varepsilon^{-O(d)} \log(n) \log^{O(d)}(\log n)$ time; here, σ^* is an optimal discrete transport plan and we assume that the spread of the supports of μ and ν is polynomially bounded.

1 Introduction

Optimal transport (OT) is a powerful tool for comparing probability distributions and computing maps between them. Put simply, the optimal transport problem deforms one distribution to the other with smallest possible cost. Classically, the OT problem has been extensively studied within the operations research, statistics, and mathematics [37, 38, 49]. In recent years, optimal transport has seen rapid rise in various machine learning and computer vision applications as a meaningful metric between distributions and has

---

*Department of Computer Science, Duke University.

†Department of Computer Science, North Carolina State University.

‡Department of Computer Science, Virginia Tech.

been extensively used in generative models [19, 25, 43], robust learning [20], supervised learning [28, 35], computer vision applications [10, 26], variational inference [6], blue noise generation [18, 41], and parameter estimation [13, 34]. These applications have led to developing efficient algorithms for OT; see the book [40] for review of computational OT.

In the *geometric OT* problem, the cost of transporting unit mass between two locations is the Euclidean distance or some L_p norm between them. In this paper, we design simple, efficient approximation algorithms for the semi-discrete and discrete geometric OT problems in fixed dimensions.

Let μ be a continuous probability distribution (i.e., density) defined over a compact bounded support $A \subset \mathbb{R}^d$, and let ν be a discrete distribution, where the support of ν, denoted by B, is a set of n points in $\mathbb{R}^d$. Let $d(\cdot, \cdot)$ be the ground metric between a pair of points in $\mathbb{R}^d$. A coupling $\tau: A \times B \to \mathbb{R}_{\geq 0}$ is called a *transport plan* for μ and ν if for all $a \subseteq A$, $\sum_{b \in B} \tau(a, b) = \mu(a)$ (where $\mu(a)$ is the mass of μ inside a) and for all $b \in B$, $\int_A \tau(a, b)\, da = \nu(b)$. The cost of the transport plan τ is given by $\mathcal{C}(\tau) := \int_A \sum_{b \in B} d(a, b)\tau(a, b)\, da$. The goal is to find a minimum-cost (semi-discrete) transport plan satisfying μ and ν[1]. For any parameter $\varepsilon > 0$, a transport plan τ between μ and ν is called ε-*close* if the cost of τ is within an additive error of ε from the cost of the optimal transport plan τ^*, i.e., $\mathcal{C}(\tau) \leq \mathcal{C}(\tau^*) + \varepsilon$. A $(1 + \varepsilon)$-*approximate OT plan*, or simply ε-*OT plan*, is a transport plan τ with $\mathcal{C}(\tau) \leq (1 + \varepsilon)\mathcal{C}(\tau^*)$.

The problem of computing semi-discrete optimal transport between μ and ν reduces to the problem of finding a set of weights $y : B \to \mathbb{R}_{\geq 0}$ so that, for any point $b \in B$, the Voronoi cell of b in the additively weighted Voronoi diagram has a mass equal to $\nu(b)$, i.e., $Vor(b) = \{x \in \mathbb{R}^d \mid d(x, b) - y(b) \leq d(x, b') - y(b'), \forall b' \in B\}$, $\mu(Vor(b)) = \nu(b)$, and the mass of μ in $Vor(b)$ is transported to b; see [9]. One can thus define an optimal semi-discrete transport plan by describing the weights of points in B. For arbitrary distributions, weights can have large bit (or algebraic) complexity, so our goal will be to compute the weights accurately up to $s = O(\log \varepsilon^{-1})$ bits, which in turn will return an ε-close semi-discrete OT plan.

If μ is also a discrete distribution with support A, a *discrete transport plan* is $\sigma: A \times B \to \mathbb{R}_{\geq 0}$ that assigns the mass transported along each edge $(a, b) \in A \times B$ such that $\sum_{b \in B} \sigma(a, b) = \mu(a)$ for each point $a \in A$ and $\sum_{a \in A} \sigma(a, b) = \nu(b)$ for each point $b \in B$. The cost of σ is given by $\mathcal{C}(\sigma) = \sum_{(a,b) \in A \times B} \sigma(a, b)d(a, b)$. The *discrete OT problem* asks for a transport plan σ with the minimum cost. We refer to such plan as an *OT plan*.

Related work. The discrete optimal transport problem under any metric can be modeled as an uncapacitated minimum-cost flow problem and can be solved in strongly polynomial time of $O((m + n \log n)n \log n)$ time using the algorithm by Orlin [39]. Using recent techniques [44], it can be solved in $n^{2+o(1)} \text{poly} \log(\Delta)$ time, where Δ depends on the spread of $A \cup B$ and the maximum demand. The special case where all points have the same demand is the widely studied *minimum-cost bipartite matching* problem. There is extensive work on the design of near-linear time approximation for the optimal transport and related matching problems [3, 7, 10, 22, 29, 42, 45]. The near-linear time algorithms by Khesin *et. al.* [29] and Fox and Lu [22] for computing an ε-OT plan use minimum-

---

[1]Apparently the semi-discrete OT was introduced by Cullen and Purser [17] without reference to optimal transport.

cost-flow (MCF) solvers (e.g. [46]) as a black box and numerically precondition their minimum-cost flow instance using geometry [22, 29, 47]. The work of Zuzic [50] describes a multiplicative-weights update (MWU) based boosting method for minimum-cost flows using an approximate primal-dual oracle as a black box, which replaces the preconditioner used in [29, 47]. All these algorithms are Monte Carlo algorithms and have running time of $n(\varepsilon^{-1}\log n)^{O(d)}$. Recently, Agarwal *et. al.* [1] presented an $n(\varepsilon^{-1}\log n)^{O(d)}$-time deterministic algorithm for computing an ε-approximate bipartite matching in $\mathbb{R}^d$. A Monte-Carlo ε-approximation algorithm for matching with run time $n\log^4 n(\varepsilon^{-1}\log\log n)^{O(d)}$ was presented in [2]. Very recently, Fox and Lu proposed a deterministic algorithm for ε-OT with run time of $O(n\varepsilon^{-(d+2)}\log^5 n\log\log n)$ [23].

The known algorithms for semi-discrete OT that compute an ε-close transport plan by and large use first and second order numerical solvers [9, 12, 16, 18, 30, 31, 33, 38]. These algorithms start with an initial set of weights for points in B and iteratively improve the weights until the mass inside the Voronoi cell of any point $b \in B$ is an additive factor ε away from $\nu(b)$. One can use these solvers to compute an ε-close transport plan by executing $\text{poly}(n, 1/\varepsilon)$ iterations. Each iteration requires computation of several weighted Voronoi diagrams which takes $n^{\Omega(d)}$ time. One can also draw samples from the continuous distribution and convert the semi-discrete OT problem to a discrete instance [24]; however, due to sampling errors, this approach provides an additive approximation. Van Kreveld *et. al.* [48] presented a $(1 + \varepsilon)$-approximation OT algorithm for the restricted case when the continuous distribution is uniform over a collection of simple geometric objects (e.g. segments, simplices, etc.), by sampling roughly n^2 points and then running an algorithm for computing discrete ε-OT mentioned above. Their running time is roughly $n^2\varepsilon^{-O(d)}\text{poly}\log(n)$.

Our contributions. We present three new algorithms for the semi-discrete and discrete optimal transport problems. Our first result is a cost-scaling algorithm that computes an ε-close transport plan for a semi-discrete instance in $n^{O(d)}\log(\Delta/\varepsilon)$ time, assuming that we have access to an oracle that, given a constant complexity region φ, returns $\mu(\varphi)$.

THEOREM 1.1. *Let μ be a continuous distribution defined on a compact bounded set $A \subset \mathbb{R}^d$, ν a discrete distribution with a support $B \subset \mathbb{R}^d$ of size n, and $\varepsilon > 0$ a parameter. Suppose there exists an ORACLE which, given a constant complexity region φ, returns $\mu(\varphi)$ in Q time. Then, an ε-close semi-discrete OT plan can be computed in $Qn^{O(d)}\log(\frac{\Delta}{\varepsilon})$ time, where Δ is the diameter of $A \cup B$.*

To the best of our knowledge, our algorithm is the first one to compute an ε-close OT in time that is polynomial in both n and $\log(\varepsilon^{-1})$. Earlier algorithms had an $\varepsilon^{-O(1)}$ factor in the run time[2]. Our algorithm not only finds the optimal transport cost within an additive error, it also finds the optimal dual weights within an additive error of ε, i.e., it computes optimal dual-weights up to $O(\log\varepsilon^{-1})$ bits of accuracy. Our algorithm works for any ground distance where the bisector of two points under the distance function $d(\cdot, \cdot)$

---

[2]Mérigot and Thibert had conjectured that an algorithm for computing an ε-close OT for semi-discrete setting with runtime $(n\log\varepsilon^{-1})^{O(1)}$ might follow using a scaling framework [36, Remark 24]. Our result proves their conjecture in the affirmative.

is an algebraic variety of constant degree. Consequently, it works for several important distances, including the L_p-norm and the squared-Euclidean distance.

The previous best-known algorithm by Kitagawa [30] for the semi-discrete optimal transport has an execution time $n^{\Omega(d)}\Delta/\varepsilon$; furthermore, their algorithm only approximates the cost and does not necessarily provide any guarantees for the optimal transport plan or the optimal dual weights of B.

For each scale δ, our algorithm starts with a set of weights assigned to B. Using these weights, it constructs an instance of the discrete optimal transport of size $n^{O(d)}$, which is then solved using a primal-dual solver. The optimal dual weights for this discrete instance are then used to refine the dual weights of B. These refined dual weights act as the starting dual weights for the next scale $\delta/2$. Starting with $\delta = \Delta$, our algorithm executes a total of $O(\log(\Delta/\varepsilon))$ scales.

Our main insight is that in scale δ, one can partition the continuous distribution μ into exponentially many regions A_δ. We prove that the dual weights and the semi-discrete transport plan τ computed by our algorithm satisfy a set of δ-optimal dual feasibility conditions (a relaxation of the classical feasibility conditions of the optimal transport), one for each $(\varrho, b) \in A_\delta \times B$, making τ a δ-close transport plan. Unfortunately, explicitly solving for τ using the partitioning A_δ will result in an exponential execution time. We overcome this difficulty by making two observations.

At the start of scale δ, we have a very good initial estimate for the dual weights of points in B from the ones computed in the previous scale. In particular, we show that there is a semi-discrete transport plan τ such that the dual feasibility constraints on every pair $(\varrho, b) \in A_\delta \times B$ with $\tau(\varrho, b) > 0$ has a slack $\le 4n\delta$. Using this claim, we show that in the optimal semi-discrete transport plan τ^*, $\tau^*(\varrho, b) = 0$ for every pair (ϱ, b) with a slack $> 4n\delta$. This allows us to restrict our attention to edges with slack $\le 4n\delta$. Unfortunately, there can be exponential number of edges with slack at most $4n\delta$. In order to overcome this difficulty, we show that all slack i edges incident on b can be compactly represented as regions between carefully constructed expansions of $O(n)$ Voronoi cells in the weighted Voronoi diagram. Using this property, we can compress the size of OT instance to $n^{O(d)}$, which can then be solved using a discrete OT solver.

We also show that by increasing the number of scales in our algorithm from $O(\log(\Delta/\varepsilon))$ to $O(\log(n\Delta/\varepsilon))$, we obtain the optimal weights on the points in B within an additive error of ε.

Next, we present another approximation algorithm for the semi-discrete setting whose running time is near-linear in n but the dependence on ε increases to $\varepsilon^{-O(d)}$.

THEOREM 1.2. *Let μ be a continuous distribution defined on a compact set $A \subset \mathbb{R}^d$, ν a discrete distribution with a support $B \subset \mathbb{R}^d$ of size n, and $\varepsilon > 0$ a parameter. Suppose there exists an* ORACLE *which, given an axis-aligned box $\square$, returns $\mu(\square)$ in Q time. Then, a $(1 + \varepsilon)$-approximate semi-discrete OT plan can be computed in $O(n\varepsilon^{-3d-2}(\log^5(n)\log(\log n) + Q))$ time. If the spread of B is polynomially bounded, a $(1 + \varepsilon)$-approximate semi-discrete OT plan can be computed in $O(n\varepsilon^{-4d-5}(\log(n)\log^{2d+5}(\log n) + Q))$ time with probability at least $\frac{1}{2}$.*

Similar to [48], the high level view of our approach is to discretize the continuous distribution and use a discrete OT algorithm. Our main contribution is a more clever sampling strategy that is more global and that works for arbitrary density (rather than

for collections of geometric objects). We prove that it suffices to sample $n\varepsilon^{-O(d)}$ points in contrast to $\Omega(n^2)$ points in [48].

Our final result is a new $(1+\varepsilon)$-approximation algorithm for the discrete transport problem.

THEOREM 1.3. *Let μ and ν be two discrete distributions with support sets $A, B \subset \mathbb{R}^d$, respectively, where $A \cup B$ is a point set of size n with polynomially bounded spread, $d \geq 1$ is a constant and $\varepsilon > 0$ a parameter. Then, a $(1+\varepsilon)$-approximate discrete OT plan between μ and ν can be computed by a Monte Carlo algorithm in $O\left(n\varepsilon^{-2d-5}\log(n)\log^{2d+5}(\log n)\right)$ time with probability at least $\frac{1}{2}$.*

As mentioned above, until recently, the best-known Monte Carlo algorithm for computing an ε-OT plan had running time $n(\varepsilon^{-1}\log n)^{O(d)}$. Recently in an independent work, Fox and Lu [23] obtained a deterministic algorithm for computing an ε-OT plan in $O(n\varepsilon^{-d-2}\log^5(n)\log(\log n))$ time. We believe that our result is of independent interest. The running time is slightly better than in [23], though of course their algorithm is deterministic. But our main contribution is a greedy primal-dual $O(\log\log n)$-approximation algorithm that is simple and geometric and runs in $O(n\log\log n)$ time. By plugging our algorithm into the multiplicative weight update method as in [50], we obtain a $(1+\varepsilon)$-approximation algorithm. We believe the derandomization technique of Lu and Fox can be applied to our algorithm, but one has to check all the technical details.

2 Computing a Highly Accurate Semi-Discrete Optimal Transport

Given a continuous distribution μ over a compact bounded set $A \subset \mathbb{R}^d$, a discrete distribution ν over a set $B \subset \mathbb{R}^d$ of n points, and a parameter $\varepsilon > 0$, we present a cost-scaling algorithm for computing an ε-close semi-discrete transport plan from μ to ν. We first describe the overall framework, then provide details of the algorithm and analyze its efficiency, and finally prove its correctness.

In our algorithm, we use a black-box primal-dual discrete OT solver PD-OT(μ', ν') that given two discrete distributions μ' and ν' defined over two point sets A' and B', returns a transport plan σ from μ' to ν' and a dual weight $y(v)$ for each point $v \in A' \cup B'$ such that for any pair $(a, b) \in A' \times B'$,

$$y(b) - y(a) \leq \mathrm{d}(a, b), \tag{2.1}$$
$$y(b) - y(a) = \mathrm{d}(a, b) \quad \text{if } \sigma(a, b) > 0. \tag{2.2}$$

Standard primal-dual methods [32] construct a transport plan while maintaining (2.1) and (2.2). For concreteness, we use Orlin's algorithm [39] that runs in $O(|A \cup B|^3)$ time.

2.1 The Scaling Framework. The algorithm works in $O(\log(\Delta\varepsilon^{-1}))$ rounds, where Δ is the diameter of $A \cup B$. In each round, we have a parameter $\delta > 0$ that we refer to as the *current scale*, and we also maintain a dual weight $y(b)$ for every point $b \in B$. Initially, in the beginning of the first round, $\delta = \Delta$ and $y(b) = 0$ for all $b \in B$. Execute the following steps $s = c\log_2(\Delta\varepsilon^{-1})$ times, where c is a sufficiently large constant[3].

---

[3]Computing an ε-close transport plan requires $O(\log(\Delta/\varepsilon))$ iterations. When the goal, on the other hand, is to obtain accurate dual weights up to $O(\log\varepsilon^{-1})$ bits, we need to execute our algorithm for $O(\log(n\Delta/\varepsilon))$ iterations. See Section 2.3.

(i) *Construct a discrete OT instance:* Using the current values of dual weights of B, as described below, construct a discrete distribution $\hat{\mu}_\delta$ with a support set X_δ, where $|X_\delta| = n^{O(d)}$, and define a (discrete) ground distance function $d_\delta : B \times X_\delta \to \{0, \ldots, 4n+1\}$.

(ii) *Solve OT instance:* Compute an optimal transport plan between discrete distributions $\hat{\mu}_\delta$ and ν using the procedure PD-OT$(\hat{\mu}_\delta, \nu)$. Let σ_δ be the coupling and $\hat{y} : B \to \mathbb{R}$ be the dual weights returned by the procedure.

(iii) *Update dual weights:* $y(b) \leftarrow y(b) + \delta \hat{y}(b)$ for each point $b \in B$.

(iv) *Update scale:* $\delta \leftarrow \delta/2$.

We refer to the jth iteration of this algorithm as *iteration j*. Our algorithm terminates when $\delta \leq \varepsilon$. We now describe the details of step (i) of our algorithm, which is the only non-trivial step. Let $y(\cdot)$ be the dual weights of B at the start of iteration j.

Constructing a discrete OT instance. We construct the discrete instance by constructing a family of Voronoi diagrams and overlaying some of their cells. For a weighted point set $P \subset \mathbb{R}^d$ with weights $w : P \to \mathbb{R}$ and a distance function $d : P \times \mathbb{R}^d \to \mathbb{R}_{\geq 0}$, we define the *weighted distance* from a point $p \in P$ to any point $x \in \mathbb{R}^d$ as $d_w(p, x) = d(p, x) - w(p)$. For a point $p \in P$, its *Voronoi cell* is $\text{Vor}_w(p) = \{x \in \mathbb{R}^d \mid d_w(p, x) \leq d_w(p', x), \forall p' \in P\}$, and the *Voronoi diagram* $\text{VD}_w(P)$ is the decomposition of $\mathbb{R}^d$ induced by Voronoi cells; see [21].

For $i \in [1, 4n+1]$ and a point $b \in B$, we define a Voronoi cell V_b^i using a weight function $w_i : B \to \mathbb{R}_{\geq 0}$, as follows. We set $w_i(b) = y(b) + i\delta$ and $w_i(b') = y(b')$ for all $b' \neq b$. We set $V_b^i = \text{Vor}_{w_i}(b)$ in $\text{VD}_{w_i}(B)$. By construction, $V_b^1 \subseteq V_b^2 \subseteq \ldots \subseteq V_b^{4n+1}$. Set $\mathcal{V}_b = \{V_b^i \mid i \in [1, 4n+1]\}$ and $\mathcal{V} = \bigcup_{b \in B} \mathcal{V}_b$ (See Figure 1(a)). Let $\mathscr{A}(\mathcal{V})$ be the *arrangement* of $\mathcal{V}$, the decomposition of $\mathbb{R}^d$ into (connected) cells induced by $\mathcal{V}$; each cell of $\mathscr{A}(\mathcal{V})$ is the maximum connected region lying in the same subset of regions of $\mathcal{V}$ [4].

For each cell φ in $\mathscr{A}(\mathcal{V})$, we choose a point r_φ arbitrarily and set its mass to $\hat{\mu}_\delta(r_\varphi) = \mu(\varphi)$, where for any region ρ in $\mathbb{R}^d$, $\mu(\rho) = \int_\rho \mu(a) \, da$ is the mass of μ inside ρ (Here we assume the mass to be 0 outside the support A of μ). Set $X_\delta = \{r_\varphi \mid \varphi \in \mathscr{A}(\mathcal{V})\}$. The resulting mass distribution on X_δ is $\hat{\mu}_\delta$.

The ground distance $d_\delta(a, b)$ between any point $b \in B$ and a point $a \in X_\delta$ is defined as

$$
d_\delta(a, b) = \begin{cases} 0, & \text{if } a \in V_b^1, \\ i, & \text{if } a \in V_b^{i+1} \setminus V_b^i, \ i \in [1, 4n], \\ 4n+1, & \text{if } a \notin V_b^{4n+1}. \end{cases}
$$

See Figure 1(b). Since each V_b^i is defined by n algebraic surfaces of constant degree, assuming the bisector of two points under the distance function $d(\cdot, \cdot)$ is an algebraic variety of constant degree, $\mathscr{A}(\mathcal{V})$ has $n^{O(d)}$ cells and a point in every cell of $\mathscr{A}(\mathcal{V})$ can be computed in $n^{O(d)}$ time [11]. Hence, $|X_\delta| = n^{O(d)}$. This completes the construction of $X_\delta, \hat{\mu}_\delta$, and d_δ.

Computing a semi-discrete transport plan. At the end of any scale δ, we compute a δ-close semi-discrete transport plan τ_δ from the discrete transport plan σ_δ as follows: For

Figure 1: (a) The i-expansions of the Voronoi cells of three points $b, b', b'' \in B$, (b) A region $\varphi \in \mathscr{A}(\mathscr{V})$ (highlighted in gray) with a representative point $r \in X_\delta$, where $d_\delta(b, r) = 0$ since $r \in V_b^1$, $d_\delta(r, b') = 1$ since $r \in V_{b'}^2 \setminus V_{b'}^1$, and $d_\delta(r, b'') = 2$ since $r \in V_{b''}^3 \setminus V_{b''}^2$ is between the 2-expansion and 3-expansion of Voronoi cell of b''. The ground distance in this figure is squared Euclidean.

any edge $(r_\varphi, b) \in X_\delta \times B$, we arbitrarily transport $\sigma_\delta(r_\varphi, b)$ mass from the points inside the region φ to the point b. A simple construction of such transport plan is to set, for any region φ, any point $a \in \varphi$, and any point $b \in B$, $\tau_\delta(a, b) = \frac{\mu(a)}{\hat{\mu}_\delta(r_\varphi)} \sigma_\delta(r_\varphi, b)$. Our algorithm will only compute the transport plan at the end of the last scale, i.e., $\delta \le \varepsilon$.

Efficiency analysis. Our algorithm runs $O(\log(\Delta \varepsilon^{-1}))$ scales, where in each scale, it constructs a discrete OT instance in $n^{O(d)}$ time and solves the OT instance using a polynomial-time primal-dual OT solver. Since the size of the discrete OT instance is $n^{O(d)}$, solving it also takes $n^{O(d)}$ time, resulting in a total execution time of $n^{O(d)} \log(\Delta \varepsilon^{-1})$ for our algorithm.

2.2 Proof of Correctness. In the discrete setting, cost scaling algorithms obtain an ε-close transport plan that satisfies (2.2) and an additive ε relaxation of (2.1). For our proof, we extend these relaxed feasibility conditions to the semi-discrete transport plan and show that, at the end of each scale δ, the semi-discrete transport plan computed by our algorithm satisfies these conditions. We use the relaxed feasibility conditions to show that our semi-discrete transport plan is δ-close. Thus, in the last scale, when $\delta \le \varepsilon$, our algorithm returns an ε-close semi-discrete transport plan from μ to ν.

δ**-optimal transport plan.** For any scale δ, we first describe a discretization of the continuous distribution into a set of regions $\mathbb{A}_\delta$ and then describe the relaxed feasibility conditions for all pairs $(\varrho, b) \in \mathbb{A}_\delta \times B$.

Consider a decomposition of the support A of the continuous distribution μ into a set of regions, where each region ϱ in the decomposition satisfies the following condition:

(P1) Assuming every point $b \in B$ has a weight $w(b)$ that is an integer multiple of δ, any two points x and y in ϱ have the same weighted nearest neighbor in B with respect to weights $w(\cdot)$,

where for any set of weights w for points in B and any point $a \in A$, we say that a point $b \in B$ is a *weighted nearest neighbor* of a if $\mathrm{d}_w(a, b) = \min_{b' \in B} \mathrm{d}_w(a, b')$. Let this set of regions be $\mathbb{A}_\delta$. For each region $\varrho \in \mathbb{A}_\delta$, let r_ϱ denote an arbitrary representative point inside ϱ.

Let $y : B \to \mathbb{R}$ denote a set of dual weights for the points in B. For each region $\varrho \in \mathbb{A}_\delta$, we derive a dual weight $y_\delta(r_\varrho)$ for its representative point as follows. Let $b_\varrho \in B$ be the weighted nearest neighbor of r_ϱ with respect to weights $y(\cdot)$. We set the dual weight of r_ϱ as

$$y_\delta(r_\varrho) \leftarrow y(b_\varrho) - \mathrm{d}(r_\varrho, b_\varrho) - \delta. \tag{2.3}$$

We say that a semi-discrete transport plan τ from μ to ν along with the set of dual weights $y(\cdot)$ for points in B is δ-*optimal* if, for each point $b \in B$ and each region $\varrho \in \mathbb{A}_\delta$,

$$y(b) - y_\delta(r_\varrho) \ \leq \ \mathrm{d}(r_\varrho, b) + \delta, \tag{2.4}$$
$$y(b) - y_\delta(r_\varrho) \ \geq \ \mathrm{d}(r_\varrho, b) \qquad \text{if } \tau(\varrho, b) > 0. \tag{2.5}$$

In the following lemma, we show that any δ-optimal transport plan $\tau, y(\cdot)$ from μ to ν is 3δ-close.

LEMMA 2.1. *Suppose $\tau, y(\cdot)$ is any δ-optimal transport plan from μ to ν and let τ^* denote any optimal transport plan from μ to ν. Then, $\mathcal{C}(\tau_\sigma) \leq \mathcal{C}(\tau^*) + \delta$.*

Let $y(\cdot)$ denote the set of dual weights maintained by our algorithm at the beginning of scale δ. For any point $b \in B$ and any region $\varrho \in \mathbb{A}_\delta$, we define a *slack* on condition (2.4) for the pair (ϱ, b), denoted by $s_\delta(\varrho, b)$, as

$$s_\delta(\varrho, b) := \left\lfloor \frac{\mathrm{d}(r_\varrho, b) + \delta - y(b) + y_\delta(r_\varrho)}{\delta} \right\rfloor \delta.$$

In the following, we describe the discretization of the continuous distribution into $\mathbb{A}_\delta$ and relate it to the discrete OT instance that is constructed in step (i) of our algorithm. Furthermore, we relate the distance d_δ computed in our algorithm to the slacks s_δ.

Discretizing the continuous distribution. Let $B = \{b_1, b_2, \ldots, b_n\}$, and let $w = \langle w_1, \ldots, w_n \rangle$ be an n-dimensional vector representing a weight assignment to the points in B. We say that the vector w is *valid* if each w_i is a non-negative integer multiple of δ and bounded by Δ. Consider the set $\mathbb{W}_\delta$ of all valid vectors, i.e., $\mathbb{W}_\delta = (\delta\mathbb{Z} \cap [0, \Delta])^n$. For a valid vector $w \in \mathbb{W}_\delta$, let $\mathrm{VD}_w(B)$ denote the weighted Voronoi diagram constructed for the points in B with weights w. The partitioning $\mathbb{A}_\delta$ is simply the overlay of all weighted Voronoi diagrams $\mathrm{VD}_w(B)$ across all valid weight vectors $w \in \mathbb{W}_\delta$ (See Figure 2).

At the beginning of scale δ, while constructing the set $\mathscr{A}(\mathscr{V})$, the dual weight of each point in B maintained by our algorithm is obtained from scale 2δ and hence, is an integer multiple of 2δ. Therefore, the Voronoi cells V_b^i of each point $b \in B$ correspond to valid weight vectors. By construction of the set $\mathbb{A}_\delta$, each region $\varrho \in \mathbb{A}_\delta$ completely lies inside some region $\varphi \in \mathscr{A}(\mathscr{V})$, i.e., each region in $\mathscr{A}(\mathscr{V})$ consists of a collection of regions in $\mathbb{A}_\delta$. In the next lemma, we establish a connection between the slacks and the distances d_δ.

Figure 2: The weighted Voronoi diagrams for four different weight vectors in $\mathbb{W}_\delta$. The ground distance in this figure is squared Euclidean.

LEMMA 2.2. *For any region $\varphi \in \mathscr{A}(\mathcal{V})$, any region $\varrho \in \mathbb{A}_\delta$ inside φ, and any point $b \in B$, if $\mathsf{d}_\delta(r_\varphi, b) \leq 4n$, then $s_\delta(\varrho, b) = \mathsf{d}_\delta(r_\varphi, b)\delta$. Furthermore, if $\mathsf{d}_\delta(r_\varphi, b) = 4n + 1$, then $s_\delta(\varrho, b) \geq (4n + 1)\delta$.*

Next, we show that for each scale δ, the semi-discrete transport plan τ_δ and dual weights $(y + \delta\hat{y})(\cdot)$ for the points in B computed by our algorithm at the end of the scale is a δ-optimal transport plan.

δ-optimality of the computed transport plan. Recall that X_δ denotes the set of representative points of the regions in $\mathscr{A}(\mathcal{V})$ and $\hat{\mu}_\delta$ is the discrete distribution over X_δ computed by our algorithm at step (i). In the following lemma, we show that any optimal transport plan σ^* from $\hat{\mu}_\delta$ to ν under distance function d_δ does not transport mass on edges $(r_\varphi, b) \in X_\delta \times B$ with cost $\mathsf{d}_\delta(r_\varphi, b) > 4n$.

LEMMA 2.3. *For any scale δ, let σ^* be any optimal transport plan from $\hat{\mu}_\delta$ to ν. For any point $b \in B$ and any region $\varphi \in \mathscr{A}(\mathcal{V})$, if σ^* transports mass from r_φ to b, then $\mathsf{d}_\delta(r_\varphi, b) \leq 4n$.*

Proof. Let $\tau_{2\delta}, y(\cdot)$ be the 2δ-optimal transport plan computed by our algorithm at scale 2δ. Let $\sigma_{2\delta}$ denote a transformation of $\tau_{2\delta}$ into a discrete transport plan from $\hat{\mu}_\delta$ to ν by simply setting, for each region $\varphi \in \mathscr{A}(\mathcal{V})$, $\sigma_{2\delta}(r_\varphi, b) := \tau_{2\delta}(\varphi, b)$. Let σ^* be any optimal transport plan from $\hat{\mu}_\delta$ to ν, where the cost of each edge (r_φ, b) is set to $\mathsf{d}_\delta(r_\varphi, b)$. Define the *residual network* $\mathscr{G}$ on the vertex set $X_\delta \cup B$ as follows. For any pair $(r, b) \in X_\delta \times B$, if $\sigma_{2\delta}(r, b) > \sigma^*(r, b)$, then we add an edge directed from b to r with a capacity $\sigma_{2\delta}(r, b) - \sigma^*(r, b)$; otherwise, if $\sigma_{2\delta}(r, b) < \sigma^*(r, b)$, then we add an edge directed from r to b with a capacity $\sigma^*(r, b) - \sigma_{2\delta}(r, b)$. This completes the construction of the residual network.

For contradiction, suppose there is a pair $(r^*, b^*) \in X_\delta \times B$ such that $\sigma^*(r^*, b^*) > 0$ and $\mathsf{d}_\delta(r^*, b^*) > 4n$. From Lemma A.5, $\sigma_{2\delta}(r^*, b^*) = 0$ since $\sigma_{2\delta}$ transports mass only on edges with distance at most 4. Hence, in the residual network $\mathscr{G}$, there is a directed edge from r^* to b^* and by Lemma A.7, the edge (r^*, b^*) is contained in a simple directed cycle

$C = \langle b_1, r_1, \ldots, b_k, r_k \rangle$ in the residual network. Define the cost of the cycle C as

$$w(C) := \sum_{\langle b,r \rangle \in C} \mathrm{d}_\delta(r,b) - \sum_{\langle r,b \rangle \in C} \mathrm{d}_\delta(r,b).$$

Since σ^* is an optimal transport plan from $\hat{\mu}_\delta$ to ν, any cycle C on the residual network have a non-negative cost. Note that the length of C is at most $2n$ since C is a simple cycle and each point of B appears at most once in C. Furthermore, by Lemma A.5, any directed edge $(b_i, r_i) \in C$ has a distance at most 4. Finally, by construction, all edges have a non-negative cost. Therefore,

$$0 \le w(C) = \sum_{\langle b,r \rangle \in C} \mathrm{d}_\delta(r,b) - \sum_{\langle r,b \rangle \in C} \mathrm{d}_\delta(r,b) \le \sum_{\langle b,r \rangle \in C} 4 - \mathrm{d}_\delta(r^*,b^*) \le 4n - \mathrm{d}_\delta(r^*,b^*) < 0,$$

which is a contradiction. Hence, σ^* cannot transport mass on edges (r^*, b^*) with cost $\mathrm{d}_\delta(r^*, b^*) > 4n$. □

Let $\sigma_\delta, \hat{y}(\cdot)$ be the optimal transport plan from $\hat{\mu}_\delta$ to ν computed at step (ii) of our algorithm, and recall that τ_δ is the transport plan from μ to ν computed at the end of scale δ. In the following lemma, we show that $\tau_\delta, (y + \delta\hat{y})(\cdot)$ is a δ-optimal transport plan.

LEMMA 2.4. *For each scale δ, let $(y + \delta\tilde{y})(\cdot)$ denote the set of dual weights for points in B computed at step (iii) of our algorithm. Then, the transport plan $\tau_\delta, (y + \delta\tilde{y})(\cdot)$ is a δ-optimal transport plan.*

Proof. Let $y_\delta(\cdot)$ denote the set of dual weights derived for the representative points of regions in $\mathbb{A}_\delta$ using Equation (2.3) at the beginning of scale δ. Consider a set of dual weights y'_δ that assigns, for each region $\varrho \in \mathbb{A}_\delta$ inside a region $\varphi \in \mathscr{A}(\mathscr{V})$, a dual weight $y'_\delta(r_\varrho) := y_\delta(r_\varrho) + \hat{y}(r_\varphi)$. First, we show that the transport plan τ_δ along with dual weights $(y + \delta\hat{y})(\cdot)$ and $y'_\delta(\cdot)$ satisfy δ-optimality conditions (2.4) and (2.5). We then show that deriving the dual weights for the representative points of the regions in $\mathbb{A}_\delta$ from the dual weights $(y + \delta\tilde{y})(\cdot)$ as in Equation (2.3) does not violate δ-optimality conditions and conclude that the transport plan τ_δ and dual weights $(y + \delta\hat{y})(\cdot)$ for points in B is δ-optimal.

For any region $\varphi \in \mathscr{A}(\mathscr{V})$, any region $\varrho \in \mathbb{A}_\delta$ inside φ, and any point $b \in B$,

- by Lemma 2.2, $\mathrm{d}_\delta(r_\varphi, b)\delta \le s_\delta(\varrho, b)$. Combining with feasibility condition (2.1),

$$\begin{aligned}
(y + \delta\tilde{y})(b) - y'_\delta(r_\varrho) &= (y(b) + \delta\tilde{y}(b)) - (y_\delta(r_\varrho) + \delta\tilde{y}(r_\varphi)) \\
&= (y(b) - y_\delta(r_\varrho)) + \delta(\tilde{y}(b) - \tilde{y}(r_\varphi)) \\
&\le (y(b) - y_\delta(r_\varrho)) + \mathrm{d}_\delta(r_\varphi, b)\delta \le (y(b) - y_\delta(r_\varrho)) + s_\delta(\varrho, b) \\
&\le (y(b) - y_\delta(r_\varrho)) + (\mathrm{d}(r_\varrho, b) - y(b) + y_\delta(r_\varrho) + \delta) \\
&= \mathrm{d}(r_\varrho, b) + \delta,
\end{aligned}$$

leading to δ-optimality condition 2.4.

- if $\tau_\delta(\varrho, b) > 0$, then σ_δ transports mass from r_φ to b, i.e., $\sigma_\delta(r_\varphi, b) > 0$. In this case, by Lemma 2.3, $d_\delta(r_\varphi, b) \leq 4n$ and by Lemma 2.2, $s_\delta(\varrho, b) = d_\delta(r_\varphi, b)\delta$. Combining with feasibility condition (2.2),

$$
\begin{aligned}
(y + \delta\tilde{y})(b) - y'_\delta(r_\varrho) &= (y(b) + \delta\tilde{y}(b)) - (y_\delta(r_\varrho) + \delta\tilde{y}(r_\varphi)) \\
&= (y(b) - y_\delta(r_\varrho)) + \delta(\tilde{y}(b) - \tilde{y}(r_\varphi)) \\
&= (y(b) - y_\delta(r_\varrho)) + d_\delta(r_\varphi, b)\delta = (y(b) - y_\delta(r_\varrho)) + s_\delta(\varrho, b) \\
&\geq (y(b) - y_\delta(r_\varrho)) + (d(r_\varrho, b) - y(b) + y_\delta(r_\varrho)) \\
&= d(r_\varrho, b),
\end{aligned}
$$

  leading to δ-optimality condition 2.5.

In Lemma A.4 in the appendix, we show that reassigning the dual weights as in Equation (2.3) does not violate δ-optimality conditions (2.4) and (2.5); hence, $\tau, (y + \delta\hat{y})(\cdot)$ is δ-optimal, as claimed. $\quad\square$

2.3 Computing Optimal Dual Weights. In this section, we show that in addition to computing an ε-close transport cost in the semi-discrete setting, our algorithm can also compute the set of dual weights for the points in B accurately, up to $O(\log \varepsilon^{-1})$ bits. To obtain such accurate set of dual weights, we execute our algorithm for $O(\log(n\Delta/\varepsilon))$ iterations so that the final value of δ when the algorithm terminates is at most $\varepsilon/5n$. In the following, we show that the dual weight computed for each point in B at the last scale is ε-close to the optimal dual weight value.

Note that any edge in the graph constructed in Step (i) of our algorithm has a cost at most $4n + 1$. Consequently, in Step (ii), the largest dual weight returned by the primal-dual solver is at most $4n + 1$[4] and in Step (iii), the dual weight of any point $b \in B$ changes by at most $(4n + 1)\delta$. Since the dual weight of b becomes the optimal dual weight in the limit, to bound the difference between the current dual weight and the optimal, it suffices if we bound the total change in the dual weights for all scales after scale $\delta \leq \varepsilon/5n$. The difference between the optimal dual weight and the current dual weight is at most

$$
(4n + 1) \sum_{i=1}^{\infty} \delta/2^i = (4n + 1)\delta \leq (4n + 1)(\varepsilon/5n) \leq \varepsilon.
$$

Therefore, after $O(\log(n\Delta/\varepsilon))$ iterations of the algorithm, the difference in the optimal dual weight $y(b)$ and the current dual weight of b is at most ε.

3 Approximation Algorithm for Semi-Discrete Optimal Transport

In this section, we present our second approximation algorithm for the semi-discrete setting that computes an ε-OT plan in $n\varepsilon^{-O(d)}\text{poly}\log(n)$ expected time. We begin by describing a few notations that help us in presenting our algorithm. Let μ, ν, A, and B be the same as above. For any point $b \in \mathbb{R}^d$ and any $r \geq 0$, let $D(b, r)$ denote the Euclidean ball of radius r centered at b. Any pair of sets $P, Q \subset \mathbb{R}^d$ is called ε-*well separated* if $\max\{\text{diam}(P), \text{diam}(Q)\} \leq \varepsilon \cdot \min_{(p,q)\in P\times Q} \|p - q\|$. Given a set S of

---

[4]Any set of dual weights returned by the algorithm can be translated by a fixed value so that the smallest dual weight becomes 0. Assuming this, it is easy to see that the largest dual weight is $4n + 1$.

n points in $\mathbb{R}^d$ and a parameter ε, a collection $W = \{(P_1, Q_1), \ldots, (P_k, Q_k)\}$ is an ε-*well separated pair decomposition* (ε-WSPD) of S if *(i)* each pair (P_i, Q_i) is ε-well separated, and *(ii)* for any distinct $p, q \in S$, there exists a pair $(P_i, Q_i) \in W$ where $p \in P_i$ and $q \in Q_i$. Given a point set $B \subset \mathbb{R}^d$ and a hypercube $\square$, we say that $\square$ is ε-*close to* $b \in B$ if $\max_{a \in \square} \|b - a\| \leq \varepsilon \min_{b' \neq b \in B} \|b' - b\|$. For any parameter $\delta > 0$, let $\mathbf{G}_\delta$ denote an axis-aligned grid of side-length δ with a vertex at the origin, i.e., $\mathbf{G}_\delta := [0, \delta]^d + \mathbb{Z}^d$. In the remainder of this section, we present our algorithm and analyze its correctness and efficiency.

3.1 Algorithm. Here is a brief overview of our algorithm. Let H be a hypercube of side-length $\frac{4}{\varepsilon}\mathrm{diam}(B)$ centered at one of the points of B. First, we partition H into a collection of hypercubes such that for each $b \in B$ and all hypercubes $\square$ except the ones that are ε-close to b, the following condition holds: for all $p, q \in \square$, $\|b - p\| \leq (1 + \varepsilon)\|b - q\|$. If a hypercube $\square$ is ε-close to $b \in B$, then we greedily route the mass of μ inside $\square$ to b. We then construct a discretization $\hat{\mu}$ of the remaining mass from μ by collapsing the mass $\mu(\square)$ of each cell $\square$ to its center point $c_\square$. We compute an ε-OT plan σ from $\hat{\mu}$ to ν using the algorithm describe in Section 4 and transform σ into a semi-discrete transport plan τ_σ by dispersing the mass transportation throughout each hypercube, as described in Section 2.1. We now describe the algorithm in more detail.

Construction of hypercubes. Let W denote an $(\frac{\varepsilon}{4})$-WSPD of B. For every pair $(B_1, B_2) \in W$, we construct a set of hypercubes closely following the construction of an approximate Voronoi diagram [8], as follows. Let $b_1 \in B_1$ and $b_2 \in B_2$ denote arbitrary representative points of B_1 and B_2, respectively. For any integer $i = 0, \ldots, t = 2\log_2(2d\varepsilon^{-1})$, define $\delta_i = 2^i \frac{\varepsilon}{2\sqrt{d}}\|b_1 - b_2\|$ and let $\mathscr{G}_i(B_1, B_2)$ denote the set of hypercubes of the grid $\mathbf{G}_{\varepsilon\delta_i}$ intersecting $D(b_1, \delta_i) \cup D(b_2, \delta_i)$. For any cell $\square \in \mathscr{G}_i(B_1, B_2)$, if there exists a child cell $\square' \subset \square$ in $\mathscr{G}_{i-1}(B_1, B_2)$, then we replace $\square$ with its 2^d child cells to keep all hypercubes interior disjoint. Set $\mathscr{G} = \bigcup_{(B_1, B_2) \in W} \bigcup_{i=0}^{t} \mathscr{G}_i(B_1, B_2)$.

Transporting local mass. For any point $b \in B$ and some sufficiently small constant $c > 0$, define its local neighborhood to be

$$\mathscr{N}_\varepsilon(b) = \left\{ \square \in \mathscr{G} : \max_{a \in \square} \|b - a\| \leq c\,\varepsilon \min_{b' \neq b} \|b' - b\| \right\}.$$

For each $b \in B$, we transport the mass locally as follows. If $\nu(b) > 0$ and there exists a hypercube $\square \subseteq \mathscr{N}_\varepsilon(b)$ with $\mu(\square) > 0$, we transport $\min\{\mu(\square), \nu(b)\}$ mass from $\square$ to b. If $\mu(\square) \leq \nu(b)$, we set $\nu(b) = \nu(b) - \mu(\square)$, delete $\square$ from $\mathscr{G}$, and repeat the above step. If $\mu(\square) > \nu(b)$, we set $\nu(b) = 0$ and scale the mass in $\square$ down so that $\mu(\square) = \mu(\square) - \nu(b)$. This process stops when either $\nu(b) = 0$ or no cell of $\mathscr{G}$ lies inside $\mathscr{N}_\varepsilon(b)$.

Discrete OT on remaining demand. Let μ' and ν' be the two distributions after transporting the local mass. Note that μ' and ν' are not necessarily probability distributions, i.e., the mass of each one of them might not add up to 1; however, the total mass in μ' equals that of ν'. Let $\mathscr{G}$ be the set of remaining hypercubes. Let $\hat{A} = \{c_\square : \square \in \mathscr{G}\} \cup \{c_0\}$ for some $c_0 \in A \setminus H$, where $c_\square$ denotes the center of $\square$. Define $\hat{\mu}(c_\square) = \int_\square \mu'(a)\,da$ for every hyper-

cube $\square \in \mathcal{G}$ and let $\hat{\mu}(c_0) = \int_A \mu'(a)da - \sum_{\square \in \mathcal{G}} \hat{\mu}(c_\square)$. We compute a $(1 + \varepsilon)$-approximate discrete transport plan σ from $\hat{\mu}$ to ν' using the algorithm described in Section 4. We then convert σ into a semi-discrete transport plan τ_σ in a straightforward manner, similar to Section 2.1. We return a transport plan $\widetilde{\tau}$ obtained from combining τ_σ with the local mass transportation committed in the previous step in a straight-forward manner. It is easy to confirm that the transport plan $\widetilde{\tau}$ is a transport plan from μ to ν. This completes the description of our algorithm.

3.2 Proof of Correctness. In this section, we show that the transport plan computed by our algorithm is a $(1 + \varepsilon)$-approximate transport plan from μ to ν. Recall that as a first step, our algorithm constructs a family $\mathcal{G}$ of hypercubes. In the following lemma, we enumerate useful properties of these hypercubes.

LEMMA 3.1. *For each $\square \in \mathcal{G}$ the hypercube $\square$ satisfies at least one of the following two conditions:*

1. *For any two points $a_1, a_2 \in \square$ and any $b \in B$, $\|a_1 - b\| \leq (1 + \varepsilon)\|a_2 - b\|$,*

2. *There exists some $b \in B$ such that $\|a - b\| \leq \varepsilon \min_{b' \neq b} \|b' - b\|$ for all $a \in \square$.*

We then use a simple triangle inequality argument similar to [5] to show that a greedy routing on $\mathcal{N}_\varepsilon(b)$ only incurs another $(1 + \varepsilon)$-relative error.

LEMMA 3.2. *Let τ^* be an optimal transport plan between μ and ν, and let $\widetilde{\tau}$ be the transport plan returned by the algorithm. There exists a transport plan $\hat{\tau}$ such that (i) $\hat{\tau} = \widetilde{\tau}$ when restricted to $\bigcup_{b \in B} \mathcal{N}_\varepsilon(b)$, and (ii) $\mathcal{c}(\hat{\tau}) \leq (1 + \varepsilon)\mathcal{c}(\tau^*)$.*

We next show that any mass outside of H can be routed arbitrarily while incurring at most $(1 + \varepsilon)$-relative error because any two points $b_1, b_2 \in B$ are approximately equidistant from any $a \in A \setminus H$.

LEMMA 3.3. *Let $\widetilde{\tau}$ be the semi-discrete transport plan constructed by our algorithm. Let τ be any arbitrary transport plan. Then,*

$$\sum_{b \in B} \int_{A \setminus \bigcup_{\square \in \mathcal{G}} \square} \|a - b\| \cdot \widetilde{\tau}(a, b) \, da \leq (1 + \varepsilon) \sum_{b \in B} \int_{A \setminus \bigcup_{\square \in \mathcal{G}} \square} \|a - b\| \cdot \tau(a, b) \, da.$$

Finally, we consider the mass that lies inside H but does not lie in a cell of $\mathcal{G}$ that is ε-close to a point of B that has survived. We use the fact that all points within such a cell $\square$ of $\mathcal{G}$ are roughly at the same distance from a point of B, i.e. for any $p, q \in \square$ where $\mu'(\square) > 0$ and for any $b \in B$ where $\nu'(b) > 0$, $\|p - q\| \leq (1 + \varepsilon)\|q - b\|$.

LEMMA 3.4. *Let $\hat{\tau}'$ be a transport plan between μ' and ν' defined by $\hat{\tau}'(a, b) = \hat{\tau}(a, b)$ if $a \notin \mathcal{N}_\varepsilon(b)$ and $\hat{\tau}'(a, b) = 0$ otherwise. Then $\mathcal{c}(\tau_\sigma) \leq (1 + \varepsilon)\mathcal{c}(\hat{\tau}')$.*

Lemmas 3.2-3.4 together imply that our algorithm returns an ε-OT plan.

LEMMA 3.5. *Let $\widetilde{\tau}$ be the transport plan computed by our algorithm, and let τ^* be an optimal transport plan between μ and ν. Then $\mathcal{c}(\widetilde{\tau}) \leq (1 + \varepsilon)\mathcal{c}(\tau^*)$.*

3.3 Efficiency analysis. Callahan and Kosaraju [14] have shown that an $(\frac{\varepsilon}{4})$-WSPD W of S of size $O(n\varepsilon^{-d})$ can be constructed in $O(n(\varepsilon^{-d} + \log n))$ time. For each pair in W, our algorithm computes $O(\log \varepsilon^{-1})$ approximate balls, where for each approximate ball, our algorithm adds $O(\varepsilon^{-d})$ hypercubes to $\mathcal{G}$. Therefore, the collection $\mathcal{G}$ of hypercubes has size $O(n\varepsilon^{-2d} \log \varepsilon^{-1})$. Hence, partitioning the hypercube H takes $O(n(\log n + \varepsilon^{-2d} \log \varepsilon^{-1}))$ time. Furthermore, computing the mass of μ inside each hypercube take $O(n\varepsilon^{-2d} \log \varepsilon^{-1} Q)$ time. Finally, note that the discrete OT instance computed by our algorithm has size $O(n\varepsilon^{-2d} \log \varepsilon^{-1})$ and hence, can be solved in $O(n\varepsilon^{-4d-5} \log(n) \log^{2d+5}(\log n) \log(\varepsilon^{-1}))$ time using the algorithm in Section 4 when the spread of B is polynomially bounded, leading to Theorem 1.2.

4 A Near-Linear ε-Approximation Algorithm for Discrete OT

In this section, we present a randomized Monte-Carlo $(1 + \varepsilon)$-approximation algorithm for the discrete OT problem. We now let μ, ν be two discrete distributions with support sets A and B, respectively, which are finite point sets in $\mathbb{R}^d$. Set $n = |A| + |B|$. We first present an overview of the algorithm, then provide details of the various steps, and finally analyze its correctness and efficiency. Our algorithm can be seen as an adaptation of the boosting framework presented by Zuzic [50] to the discrete optimal transport problem; we present an $O(\log \log n)$-approximation algorithm for the discrete OT problem and then boost the accuracy of our algorithm using the multiplicative weights update method and compute a $(1 + \varepsilon)$-approximate discrete OT plan.

4.1 Overview of the Algorithm. At a high level, we compute a hierarchical graph $\mathcal{G} = (V, E)$, where $V \supseteq A \cup B$ is a set of points in $\mathbb{R}^d$. The weight of an edge is the Euclidean distance between its endpoints. The construction of $\mathcal{G}$ is randomized, and $\mathcal{G}$ is a $(1 + \varepsilon)$-*spanner* in expectation, i.e., $d_{\mathcal{G}}(a, b)$, the shortest-path distance between $(a, b) \in A \times B$ in $\mathcal{G}$ satisfies the condition $\| - \| \leq E[d_{\mathcal{G}}(a, b)] \leq (1 + \varepsilon)\|a - b\|$. We formulate the OT problem as a min-cost flow problem in $\mathcal{G}$ by setting $\eta(u) = \mu(u)$ if $u \in A$ and $\eta(u) = -\nu(u)$ if $u \in B$. Following a bottom-up greedy approach, we construct a flow $\sigma : V \to \mathbb{R}_{\geq 0}$ and dual weights $y : V \to \mathbb{R}$ that satisfy (C1) and (C2) with $\rho = a_1 \log \log n$, where $a_1 > 0$ is a constant:

(C1) $|y(u) - y(v)| \leq \rho \|u - v\| \quad \forall (u, v) \in E$,

(C2) $\sum_{(u,v) \in E} \sigma(u, v)\|u - v\| \leq \sum_{u \in V} y(u)\eta(u)$.

The first condition guarantees the dual solution y is ρ-approximately feasible, while the second condition guarantees that y is non-trivial and the flow σ is a ρ-approximation. Using such a primal-dual solution, one can use multiplicative-weight-update method (MWU) to boost a ρ-approximate flow into a $(1 + \varepsilon)$-approximate flow on $\mathcal{G}$ by making $O(\rho^2 \varepsilon^{-2} \log n)$ calls to our greedy primal-dual approximation algorithm. We also describe the multiplicative weights procedure in Section 4.4. Once a $(1 + \varepsilon)$-approximate flow is obtained in $\mathcal{G}$, then one can simply shortcut paths in $\mathcal{G}$ to obtain an ε-OT plan; see e.g. [23].

We remark that a $(1 + \varepsilon)$-spanner is not needed if only a $O(\log \log n)$-approximation is desired. An $O(\log \log n)$-OT plan can be constructed directly in $O(n \log \log n)$ time using our algorithm. We now describe the details of our algorithm.

4.2 Constructing a spanner. We now define the construction of the graph $\mathcal{G}$, which is built upon a hierarchical partitioning of $\mathbb{R}^d$ and the tree $\mathcal{T}$ associated to it.

Hierarchical partitioning. For simplicity, we refer to all d-dimensional hypercubes as cells. For any cell $\square$, let $\ell_\square$ and $c_\square$ denote its side-length and center, respectively. Let $\Delta = \frac{\max_{p,q \in A \cup B} \|p - q\|}{\min_{p,q \in A \cup B} \|p - q\|}$ denote the *spread* of $A \cup B$. Additionally, define $\mathbb{G}(\square, \ell)$ to be the grid that partitions $\square$ into new cells of side-length ℓ. Without loss of generality, assume $A \cup B \subseteq [0, \Delta]^d$.

Let $\square^*$ be a randomly shifted cell of side-length 2Δ containing all points in $A \cup B$, i.e., $\square^* = [0, 2\Delta]^d - x$ for some x chosen uniformly at random from the hypercube $[0, \Delta]^d$. We construct a hierarchical partition of $\square^*$ as follows. We designate $\square^*$ as the root cell of $\mathcal{T}$. For any cell $\square$ of $\mathcal{T}$, define $n_\square := |(A \cup B) \cap \square|$ as the number of points of $A \cup B$ contained within $\square$. We construct $\mathcal{T}$ recursively as follows. If $n_\square \leq \left(\varepsilon^{-1} \log \log n\right)^{3d}$, $\square$ is a leaf of $\mathcal{T}$. Otherwise, using the grid $\mathbb{G}_\square = \mathbb{G}\left(\square, \ell_\square / n_\square^{\frac{1}{3d}}\right)$, we partition $\square$ into smaller cells of side-length $\ell_\square / n_\square^{\frac{1}{3d}}$. We add all non-empty cells of $\mathbb{G}_\square$ to $\mathcal{T}$ as the children of $\square$ and denote them by $\mathsf{C}[\square]$. The height h of $\mathcal{T}$ is $h = O(\log \log n)$.

For any cell $\square$ of $\mathcal{T}$, we define a set of $O((\varepsilon^{-1} dh)^d)$ equal-sized subcells as follows. Define $\delta_\square = \frac{\varepsilon \ell_\square}{4dh}$ to be the side-length of the subcells of $\square$. We add all the cells of the grid $\mathbb{G}(\square, \delta_\square)$ that contain a point of $A \cup B$ as the subcells of $\square$ and denote the resulting family by $\mathsf{S}[\square]$.

Vertices and edges of the graph. The vertex set of $\mathcal{G}$ consists of the points $A \cup B$ plus the center point of all non-empty cells and subcells of $\mathcal{T}$. More precisely,

$$V = (A \cup B) \cup \bigcup_{\square \in \mathcal{T}} \{c_\square\} \cup \{c_\xi : \xi \in \mathsf{S}[\square]\}.$$

The edge set of $\mathcal{G}$ consists of two sets of edges per cell of $\mathcal{T}$.

1. If $\square$ is a non-leaf cell, let $\mathscr{I}_\square = c_\square \cup \left(\bigcup_{\square' \in \mathsf{C}[\square]} c_{\square'}\right) \cup \left(\bigcup_{\xi \in \mathsf{S}[\square]} c_\xi\right)$ be the set of points composed of the center of $\square$, centers of its children, and the centers of the subcells of $\square$. Otherwise, let $\mathscr{I}_\square = c_\square \cup ((A \cup B) \cap \square)$. We construct a $(1 + \varepsilon)$-spanner $\mathscr{S}_\square$ on $\mathscr{I}_\square$. We add all edges of $\mathscr{S}_\square$ to $\mathcal{G}$ and refer to them as *greedy edges*. Note that $|\mathscr{I}_\square| = |\mathsf{C}[\square]| + |\mathsf{S}[\square]| + 1 = O(n_\square^{1/3} + (h/\varepsilon)^d)$ for any non-leaf cell.

2. In addition, for any non-leaf cell $\square$, let $X_\square = \bigcup_{\square' \in \mathsf{C}[\square]} \bigcup_{\xi \in \mathsf{S}[\square']} c_\xi$ be the set of centers of the subcells of the children of $\square$. Let $\mathscr{S}'_\square$ be a $(1 + \varepsilon)$-spanner constructed on the points in $X_\square$. We add all the edges of $\mathscr{S}'_\square$ to $\mathcal{G}$ and refer to them as *shortcut edges*.

Recall that the weight of every edge in $\mathcal{G}$ is the Euclidean distance between its endpoints. The greedy edges are the edges that our greedy algorithm uses to compute a flow, whereas the shortcut edges guarantee that the shortest-path distances in $\mathcal{G}$ are a $(1 + \varepsilon)$-approximation of the Euclidean distances in expectation. We remark that the

Figure 3: The hierarchical structure of the graph $\mathscr{G}$. The vertices of the graph are the centers of the cells (blue disks) and centers of subcells (red squares). For any cell $\square$, the greedy edges form a spanner on its children and subcells (black triangles) and the shortcut edges form a spanner on the center of the subcells of its children (purple dashed rectangle).

shortcut edges are only necessary when applying the MWU method to obtain a $(1 + \varepsilon)$-approximate transport plan, otherwise only greedy edges are necessary for a $O(\log \log n)$-approximation.

For any pair $(a, b) \in A \times B$, let $P_{a,b}$ be the shortest path in $\mathscr{G}$ from a to b with respect to Euclidean distances along each edge and $\phi(P_{a,b})$ to be the cost of $P_{a,b}$, i.e. the sum of Euclidean distances of every edge in $P_{a,b}$. The following lemma bounds the size of $\mathscr{G}$ and shows that the shortest path metric of $\mathscr{G}$, in expectation, $(1 + \varepsilon)$-approximates Euclidean distances.

LEMMA 4.1. *The graph $\mathscr{G}$ contains $O(nh)$ vertices and $O(n\varepsilon^{-d}h)$ edges. The max degree of any vertex in $\mathscr{G}$ is at most $O(\varepsilon^{-d} \log n)$. Furthermore, for any pair of points (a, b), $\phi(P_{a,b}) \geq \|a - b\|$ and $\mathbb{E}\left[\phi(P_{a,b})\right] \leq (1 + 3\varepsilon)\|a - b\|$.*

4.3 Greedy Primal-Dual Algorithm. Given the graph $\mathscr{G} = (V, E)$ and a demand function $\eta \colon V \to \mathbb{R}$, we compute a flow σ on $\mathscr{G}$ satisfying the demand function η and a set of dual weights y satisfying the conditions (C1) and (C2) with a parameter $\rho = a_1 \varepsilon^{-1} \log \log n$, where a_1 is a constant depending on d. It transports as much demand as possible among children of each cell, and routes all excess up the tree $\mathscr{T}$. Due to the high branching factor of the cells in $\mathscr{T}$, each subcell contains polynomially many child-subcells. Therefore, subcells cannot simply inherit the dual weights from cells as in [29], since it might violate condition (C1). Instead, we create a min-cost flow instance for each cell consisting of the centers of its immediate descendants and compute a primal-dual flow on this instance.

Dual assignment and flow function. We now compute the primal-dual pair (σ, y) in a bottom-up manner. At any cell $\square$ we assume that all excess mass has been routed to $c_{\square'}$ for each child $\square' \in C[\square]$, and then route all excess mass from the children of $\square$ to $c_\square$. We denote the value of this excess demand in a subtree rooted at $\square$ as $\bar{\eta}_\square$, and it is

defined as follows. If $\square$ is a leaf cell, then $\bar{\eta}_\square = \eta(c_\square) + \sum_{p \in (A \cup B) \cap \square} \eta(p)$. Otherwise, $\bar{\eta}_\square = \eta(c_\square) + \sum_{\square' \in C[\square]} \bar{\eta}_{\square'} + \sum_{\xi \in S[\square]} \bar{\eta}_\xi$.

We wish to run Orlin's primal-dual algorithm for min-cost flow on $\mathscr{S}_\square$ [39]. However, we only assume that η is a balanced demand function on the whole vertex set of $\mathscr{G}$. The total mass in $\mathscr{S}_\square$ defined by η may not be balanced on some subgraph $\mathscr{S}_\square$. To resolve this issue, we make $c_\square$ a sink node that absorbs all excess mass from η in the subgraph rooted at $\square$. We define a local demand function $\eta_\square : \mathscr{I}_\square \to \mathbb{R}$ as follows. For each child $\square' \in C[\square]$, $\eta_\square(c_{\square'}) = \bar{\eta}_{\square'}$, for each subcell $\xi \in S[\square]$, $\eta_\square(c_\xi) = \eta(c_\xi)$, and,

$$\eta_\square(c_\square) = - \sum_{\square' \in C[\square]} \eta_\square(c_{\square'}) - \sum_{\xi \in S[\square]} \eta_\square(c_\xi).$$

Roughly speaking, the demand at the center of a child node $\square'$ is the surplus/deficit in the subtree rooted at $\square'$. The demand at the center of $\square$ is set so that the net excess of demands in $\mathscr{I}_\square$ is rooted to $c_\square$ and similarly, the net deficit of $\mathscr{I}_\square$ is supplied from $c_\square$. The pair $(\mathscr{S}_\square, \eta_\square)$ is a balanced instance for the min-cost flow. We now run Orlin's primal-dual algorithm for uncapacitated minimum-cost flow to obtain a local primal-dual pair $(\sigma_\square, y_\square)$ on $(\mathscr{S}_\square, \eta_\square)$ [39]. The combination of all flows computed at all cells of $\mathscr{T}$ satisfies the demand function η.

Suppose $(\sigma_\square, y_\square)$ is the primal-dual flow computed on the local instance $(\mathscr{S}_\square, \eta_\square)$,. For any point $u \in \mathscr{I}_\square$, we define the dual weight of u as $y(u) \leftarrow y_\square(u) - y_\square(c_\square) + y(c_\square)$. The definition of y synchronizes all the local dual weights computed for each cell of the tree. Additionally, observe that each edge (u, v) of $\mathscr{G}$ belongs to a unique local instance $(\mathscr{S}_\square, \eta_\square)$ of min cost flow. We simply define $\sigma(u, v) = \sigma_\square(u, v)$, where $\square$ is the cell for which (u, v) is contained in $\mathscr{I}_\square$. This completes the construction of our greedy primal-dual algorithm.

4.4 Multiplicative Weights Update (MWU) Framework.

Using one of the known algorithms [15, 27], we first compute an estimate of the OT cost within a $d \log n$ factor in $O(n \log n)$ time, i.e. we compute a value $\tilde{g}$ such that $w^* \leq \tilde{g} \leq (d \log n) \cdot w^*$. Using this estimate, we perform an exponential search in the range $\left[\frac{\tilde{g}}{d \log n}, \tilde{g} \right]$ with increments of factor $(1 + \varepsilon)$. For any guess value g, the MWU algorithm either returns a flow $\sigma \colon E \to \mathbb{R}$ with $\text{¢}(\sigma) \leq (1 + \varepsilon)g$ or returns dual weights as a certificate that $g < w^*$. We now describe the MWU algorithm for a fixed value of g.

Set $T = 4\rho^2 \varepsilon^{-2} \log |E|$. The algorithm runs in at most T iterations, where in each iteration, it maintains a pre-flow vector σ^t satisfying $\text{¢}(\sigma^t) \leq g$. The pre-flow σ^t need not route all demand successfully. Initially, set $\sigma^0(u, v) = \frac{g}{\|u - v\| \cdot |E|}$ for each edge $(u, v) \in E$. For each iteration t, define the *residual demand* $\eta_{\text{res}}^t(\cdot)$ as

$$\eta_{\text{res}}^t(u) = \eta(u) - \sum_{v : (u,v) \in E} (\sigma^{t-1}(u, v) - \sigma^{t-1}(v, u)).$$

Let $(\sigma_{\text{res}}^t, y^t)$ be the primal-dual flow computed by our greedy algorithm for the residual demands η_{res}^t. Recall that $(\sigma_{\text{res}}^t, y^t)$ satisfies (C1) and (C2). If $\langle \eta_{\text{res}}^t, y^t \rangle \leq \varepsilon g$, then (C2) implies that $\text{¢}(\sigma_{\text{res}}^t) \leq \varepsilon g$. Since σ_{res}^t routes the residual demands, the flow function $\sigma^t = \sigma^{t-1} + \sigma_{\text{res}}^t$ routes the original demand η with a cost $\text{¢}(\sigma^t) \leq (1 + \varepsilon)g$. In this case, the algorithm returns σ^t as the desired flow and terminates.

Otherwise, $\langle \eta_{\text{res}}^t, y^t \rangle > \varepsilon g$ and we update the flow along each edge $e = (u,v)$ of G based on the slack $s^t(u,v) = \frac{y^t(u) - y^t(v)}{\|u-v\|}$ of e with respect to dual weights y^t:

$$\sigma^t(u,v) \leftarrow \exp\left(\frac{\varepsilon}{2\rho^2} s^t(u,v)\right) \cdot \sigma^{t-1}(u,v).$$

We emphasize that flow along an edge is increasing if the slack is large. Then, one needs to rescale σ^t so that its cost is bounded above by g. If the algorithm does not terminate within T rounds, we conclude that the value of g is an under-estimate of the cost of the min-cost flow; we increase g by a factor of $(1+\varepsilon)$ and repeat the MWU algorithm. This completes the description of the MWU framework.

4.5 Analysis. The following two lemmas prove that our algorithm satisfies conditions (C1) and (C2) for a sufficiently small approximation factor.

LEMMA 4.2. *For any edge* $(u,v) \in E$, $|y(u) - y(v)| \leq O(d^{3/2} h \varepsilon^{-1}) \|u - v\|$.

LEMMA 4.3. $\sum_{(u,v) \in E} \sigma(u,v) \|u - v\| \leq \sum_{u \in V} y(u) \eta(u)$.

Next, we bound the running time of our algorithm. For any cell $\square$, the algorithm computes an exact primal-dual solution to min-cost flow on $\mathscr{I}_\square$ with demands $\eta_\square(\cdot)$ in $O(|\mathscr{I}_\square|^3)$ time. Each cell $\square$ satisfies $|\mathscr{I}_\square| = O\left(n_\square^{1/3} + (h/\varepsilon)^d\right)$. The total number of points inside the cells of level i is n; i.e., $\sum_{\square \in \mathscr{L}[i]} n_\square = n$. Furthermore, the total number of non-empty subcells of the cells at level i is at most n; i.e., $\sum_{\square \in \mathscr{L}[i]} O(h/\varepsilon)^d \leq n$. Therefore,

$$\sum_{\square \in \mathscr{L}[i]} |\mathscr{I}_\square|^3 = \sum_{\square \in \mathscr{L}[i]} O\left(n_\square + \left(h\varepsilon^{-1}\right)^{3d}\right) = O\left(n\left(h\varepsilon^{-1}\right)^{2d}\right).$$

Summing over all levels of $\mathscr{T}$, the total running time of the algorithm is $\tilde{O}\left(n\left(h/\varepsilon\right)^{2d+1}\right)$.

Acknowledgement

References

[1] P. K. Agarwal, H.-C. Chang, S. Raghvendra, and A. Xiao. Deterministic, near-linear ε-approximation algorithm for geometric bipartite matching. In *Proc. 54th Annual ACM Sympos. on Theory of Comput.*, pages 1052–1065, 2022.

[2] P. K. Agarwal, S. Raghvendra, P. Shirzadian, and R. Sowle. An improved ε-approximation algorithm for geometric bipartite matching. In *Proc. 18th Scandinavian Sympos. and Workshops Algorithm Theory*, 2022.

[3] P. K. Agarwal and R. Sharathkumar. Approximation algorithms for bipartite matching with metric and geometric costs. *In Proc. Forty-Sixth annual ACM Sympos. on Theory of Comput.*, page 555–564, 2014.

[4] P. K. Agarwal and M. Sharir. Efficient algorithms for geometric optimization. *ACM Comput. Surveys (CSUR)*, 30(4):412–458, 1998.

[5] P. K. Agarwal and K. R. Varadarajan. A near-linear constant-factor approximation for Euclidean bipartite matching? In *20th Annual Sympos. on Comput. Geometry*, pages 247–252, 2004.

[6] L. Ambrogioni, U. Guclu, and M. van Gerven. Wasserstein variational gradient descent: From semi-discrete optimal transport to ensemble variational inference. *arXiv preprint arXiv:1811.02827*, 2018.

[7] A. Andoni, P. Indyk, and R. Krauthgamer. Earth mover distance over high-dimensional spaces. In *SODA*, volume 8, pages 343–352, 2008.

[8] S. Arya and T. Malamatos. Linear-size approximate voronoi diagrams. In *SODA*, pages 147–155, 2002.

[9] F. Aurenhammer, F. Hoffmann, and B. Aronov. Minkowski-type theorems and least-squares clustering. *Algorithmica*, 20(1):61–76, 1998.

[10] A. Backurs, Y. Dong, P. Indyk, I. Razenshteyn, and T. Wagner. Scalable nearest neighbor search for optimal transport. *In International Conference on Machine Learning*, pages 497–506, 2020.

[11] S. Basu, R. Pollack, and M. Roy. Algorithms in real algebraic geometry. algorithms and computat, 2003.

[12] J.-D. Benamou and Y. Brenier. A computational fluid mechanics solution to the monge-kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, 2000.

[13] E. Bernton, P. E. Jacob, M. Gerber, and C. P. Robert. On parameter estimation with the wasserstein distance. *Information and Inference: A Journal of the IMA*, 8(4):657–676, 2019.

[14] P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the ACM (JACM)*, 42(1):67–90, 1995.

[15] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proc. thiry-fourth annual ACM Sympos. on Theory of Comput.*, pages 380–388, 2002.

[16] R. Chartrand, B. Wohlberg, K. Vixie, and E. Bollt. A gradient descent solution to the monge-kantorovich problem. *Applied Mathematical Sciences*, 3(22):1071–1080, 2009.

[17] M. J. Cullen and R. J. Purser. An extended lagrangian theory of semi-geostrophic frontogenesis. *Journal of Atmospheric Sciences*, 41(9):1477–1497, 1984.

[18] F. De Goes, K. Breeden, V. Ostromoukhov, and M. Desbrun. Blue noise through optimal transport. *ACM Transactions on Graphics (TOG)*, 31(6):1–11, 2012.

[19] I. Deshpande, Z. Zhang, and A. G. Schwing. Generative modeling using the sliced wasserstein distance. In *Proc. IEEE conference on computer vision and pattern recognition*, pages 3483–3491, 2018.

[20] P. M. Esfahani and D. Kuhn. Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1):115–166, 2018.

[21] S. Fortune. Voronoi diagrams and delaunay triangulations. *Comput. in Euclidean geometry*, pages 225–265, 1995.

[22] K. Fox and J. Lu. A near-linear time approximation scheme for geometric transportation with arbitrary supplies and spread. In *Proc. 36th Annual Sympos. on Comput. Geometry*, pages 45:1–45:18, 2020.

[23] K. Fox and J. Lu. A deterministic near-linear time approximation scheme for geometric transportation. *arXiv preprint arXiv:2211.03891*, 2022.

[24] A. Genevay, M. Cuturi, G. Peyré, and F. Bach. Stochastic optimization for large-scale optimal

transport. *Advances in neural information processing systems*, 29, 2016.

[25] A. Genevay, G. Peyre, and M. Cuturi. Learning generative models with sinkhorn divergences. *In International Conference on Artificial Intelligence and Statistics*, page 1608–1617, 2018.

[26] R. Gupta, P. Indyk, and E. Price. Sparse recovery for earth mover distance. In *2010 48th Annual Allerton Conference on Communication, Control, and Comput. (Allerton)*, pages 1742–1744. IEEE, 2010.

[27] P. Indyk and N. Thaper. Fast image retrieval via embeddings. In *3rd international workshop on statistical and Comput. theories of vision*, volume 2, page 5, 2003.

[28] H. Janati, M. Cuturi, and A. Gramfort. Wasserstein regularization for sparse multi-task regression. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1407–1416. PMLR, 2019.

[29] A. B. Khesin, A. Nikolov, and D. Paramonov. Preconditioning for the geometric transportation problem. *arXiv preprint arXiv:1902.08384*, 2019.

[30] J. Kitagawa. An iterative scheme for solving the optimal transportation problem. *Calculus of Variations and Partial Differential Equations*, 51(1):243–263, 2014.

[31] J. Kitagawa, Q. Mérigot, and B. Thibert. Convergence of a newton algorithm for semi-discrete optimal transport. *Journal of the European Mathematical Society*, 21(9):2603–2651, 2019.

[32] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[33] B. Lévy and E. L. Schwindt. Notions of optimal transport theory and how to implement them on a computer. *Computers & Graphics*, 72:135–148, 2018.

[34] H. Liu, G. U. Xianfeng, and D. Samaras. A two-step computation of the exact gan wasserstein distance. *In International Conference on Machine Learning*, pages 3159–3168, 2018.

[35] G. Luise, A. Rudi, M. Pontil, and C. Ciliberto. Differential properties of sinkhorn approximation for learning with wasserstein distance. *Advances in Neural Information Processing Systems*, 31, 2018.

[36] Q. Merigot and B. Thibert. Optimal transport: discretization and algorithms. In *Handbook of numerical analysis*, volume 22, pages 133–212. Elsevier, 2021.

[37] J.-M. Mirebeau. Discretization of the 3d monge- ampere operator, between wide stencils and power diagrams. *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*, 49(5):1511–1523, 2015.

[38] V. I. Oliker and L. D. Prussner. On the numerical solution of the equation $\frac{\partial^2 z}{\partial x^2} \frac{\partial^2 z}{\partial y^2} - \left(\frac{\partial^2 z}{\partial x \partial y}\right) = f$ and its discretizations, i. *Numerische Mathematik*, 54(3):271–293, 1989.

[39] J. Orlin. A faster strongly polynomial minimum cost flow algorithm. In *Proc. Twentieth annual ACM Sympos. on Theory of Comput.*, pages 377–387, 1988.

[40] G. Peyré, M. Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[41] H. Qin, Y. Chen, J. He, and B. Chen. Wasserstein blue noise sampling. *ACM Transactions on Graphics (TOG)*, 36(5):1–13, 2017.

[42] S. Raghvendra and P. K. Agarwal. A near-linear time ε-approximation algorithm for geometric bipartite matching. *Journal of the ACM (JACM)*, 67(3):1–19, 2020.

[43] T. Salimans, H. Zhang, A. Radford, and D. Metaxas. Improving gans using optimal transport. *In International Conference on Learning Representations*, 2018.

[44] R. Seshadri and K. K. Srinivasan. Algorithm for determining path of maximum reliability on a network subject to random arc connectivity failures. *Transportation Research Record*, 2467(1):80–90, 2014.

[45] R. Sharathkumar and P. K. Agarwal. Algorithms for the transportation problem in geometric settings. In *Proc. 23rd annual ACM-SIAM Sympos. on Discrete Algorithms*, pages 306–317. SIAM, 2012.

[46] J. Sherman. Generalized preconditioning and undirected minimum-cost flow. In *Proc.*

Twenty-Eighth Annual ACM-SIAM Sympos. on Discrete Algorithms, pages 772–780, 2017.

[47] J. Sherman. Generalized preconditioning and undirected minimum-cost flow. In *Proc. Twenty-Eighth Annual ACM-SIAM Sympos. on Discrete Algorithms*, pages 772–780. SIAM, 2017.

[48] M. van Kreveld, F. Staals, A. Vaxman, and J. Vermeulen. Approximating the earth mover's distance between sets of geometric objects. *arXiv preprint arXiv:2104.08136*, 2021.

[49] C. Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

[50] G. Zuzic. A simple boosting framework for transshipment. *arXiv preprint arXiv:2110.11723*, 2021.

A Missing Details and Proofs of Section 2

In this section, we present the missing details and the proofs of the claims made in Section 2.

A.1 Weighted Nearest Neighbor. Let $w : B \to \mathbb{R}^{\geq 0}$ denote a set of non-negative weights for the points in B. Recall that for any pair of points $(a, b) \in A \times B$, the weighted distance of a and b with respect to w is $d_w(a, b) = d(a, b) - w(b)$. For any point $a \in A$, the *weighted nearest neighbor (WNN)* of a is a point $b \in B$ with the smallest weighted distance to a, i.e, a point $b \in B$ satisfying $d_w(a, b) = \min_{b' \in B} d_w(a, b')$. For any $\delta > 0$ and any point $a \in A$, we say that a point $b \in B$ is a *δ-approximate weighted nearest neighbor (δ-WNN)* of a if $d_w(a, b) \leq \min_{b' \in B} d_w(a, b') + \delta$.

LEMMA A.1. *Given a transport plan τ from μ to ν and a parameter $\delta > 0$, suppose there exists a set of weights w for the points in B such that for any pair of points $(a, b) \in A \times B$ with $\tau(a, b) > 0$, the point b is a δ-WNN of a with respect to weights w. Then, τ is a δ-close transport plan from μ to ν.*

Proof. For any transport plan τ', we define the weighted cost of τ', denoted by $\mathfrak{c}_w(\tau')$, as the cost of the τ' where the edge costs are replaced with the weighted distance between the points, i.e., $\mathfrak{c}_w(\tau') := \sum_{b \in B} \int_A d_w(a, b) \tau'(a, b) \, da$. For any transport plan τ',

$$
\begin{aligned}
\mathfrak{c}_w(\tau') &= \sum_{b \in B} \int_A d_w(a, b) \tau'(a, b) \, da = \sum_{b \in B} \int_A (d(a, b) - w(b)) \tau'(a, b) \, da \\
&= \sum_{b \in B} \int_A d(a, b) \tau'(a, b) \, da - \sum_{b \in B} w(b) \int_A \tau'(a, b) \, da \\
&= \mathfrak{c}(\tau') - \sum_{b \in B} w(b) \nu(b).
\end{aligned}
\tag{A.1}
$$

For any point $a \in A$, suppose b_a denotes any WNN of a. Furthermore, for any point $a \in A$, let $\mathscr{M}_\tau(a)$ denote the set of all points $b \in B$ such that $\tau(a, b) > 0$. Let τ^* denote any optimal transport plan from μ to ν.

$$
\begin{aligned}
\mathfrak{c}_w(\tau) &= \int_A \sum_{b \in \mathscr{M}_\tau(a)} d_w(a, b) \tau(a, b) \, da \leq \int_A \sum_{b \in \mathscr{M}_\tau(a)} (d_w(a, b_a) + \delta) \tau(a, b) \, da \\
&= \delta + \int_A d_w(a, b_a) \mu(a) \, da \leq \delta + \int_A \sum_{b \in B} d_w(a, b) \tau^*(a, b) \, da \\
&= \mathfrak{c}_w(\tau^*) + \delta.
\end{aligned}
\tag{A.2}
$$

Combining Equations (A.1) and (A.2),

$$
\mathfrak{c}(\tau) = \mathfrak{c}_w(\tau) + \sum_{b \in B} w(b) \nu(b) \leq \mathfrak{c}_w(\tau) + \delta + \sum_{b \in B} w(b) \nu(b) = \mathfrak{c}(\tau^*) + \delta,
$$

i.e., the transport plan τ is a δ-close transport plan. ∎

A.2 δ-Optimal Transport Plan. Given a continuous distribution μ defined over a compact bounded set A, a discrete distribution ν defined on a point set B, and a parameter $\delta > 0$, recall that $\mathcal{A}_\delta$ denotes a partitioning over the set A, which is the arrangement of all weighted Voronoi diagrams $\mathrm{VD}_w(B)$ for all valid weight vectors $w \in \mathbb{W}_\delta$. Recall that for each region $\varrho \in \mathcal{A}_\delta$, we refer to its representative point by r_ϱ. In the following lemma, we show an important property of the partitioning $\mathcal{A}_\delta$.

LEMMA A.2. *For any region $\varrho \in \mathcal{A}_\delta$, any pair of points $a_1, a_2 \in \varrho$, and any valid weight vector $w \in \mathbb{W}_\delta$, any δ-WNN of a_1 is also a δ-WNN for a_2.*

Proof. Suppose a point $b \in B$ is a δ-WNN of the point a_1, i.e., for any point $b' \in B$,

$$\mathrm{d}_w(a_1, b) - \delta \leq \mathrm{d}_w(a_1, b'). \tag{A.3}$$

Define the weights $w_+(\cdot)$ as a set of weights that assigns $w_+(b) = w(b) + \delta$ and $w_+(b') = w(b')$ to each point $b' \neq b$ in B. Note that w_+ is also a valid weight vector. For any point $b' \neq b$ in B, by Equation (A.3),

$$\mathrm{d}_{w_+}(a_1, b) = \mathrm{d}(a_1, b) - w_+(b) = \mathrm{d}(a_1, b) - w(b) - \delta = \mathrm{d}_w(a_1, b) - \delta \leq \mathrm{d}_w(a_1, b')$$
$$= \mathrm{d}_{w_+}(a_1, b'). \tag{A.4}$$

In other words, b is a WNN for the point a_1 with respect to weights w_+. Since $w_+ \in \mathbb{W}_\delta$, by the construction of $\mathcal{A}_\delta$, the region ϱ completely lies inside the Voronoi cell of b in the weighted Voronoi diagram $\mathrm{VD}_{w_+}(B)$. As a result, b is also a WNN for the point a_2 with respect to the weights $w_+(\cdot)$. Therefore, for any point $b' \neq b$ in B,

$$\mathrm{d}_w(a_2, b) - \delta = \mathrm{d}(a_2, b) - w(b) - \delta = \mathrm{d}_{w_+}(a_2, b) \leq \mathrm{d}_{w_+}(a_2, b') = \mathrm{d}_w(a_2, b'),$$

i.e., the point b is also a δ-WNN for a_2. □

Lemma A.3 follows from combining Lemmas A.1 and A.2 in a straight-forward way.

LEMMA A.3. *Suppose τ is a transport plan from μ to ν and $w \in \mathbb{W}_\delta$ a valid weight vector such that for any pair $(\varrho, b) \in \mathcal{A}_\delta \times B$ with $\tau(\varrho, b) > 0$, the point b is a δ-WNN of r_ϱ. Then, τ is a δ-close transport plan from μ to ν.*

In the following lemma, we show that any δ-optimal transport plan $\tau, y(\cdot)$ from μ to ν is δ-close.

LEMMA 2.1. *Suppose $\tau, y(\cdot)$ is any δ-optimal transport plan from μ to ν and let τ^* denote any optimal transport plan from μ to ν. Then, $\mathcal{c}(\tau_\sigma) \leq \mathcal{c}(\tau^*) + \delta$.*

Proof. To prove this lemma, we first show that for any pair $(\varrho, b) \in \mathcal{A}_\delta \times B$ such that $\tau(\varrho, b) > 0$, the point b is a δ-WNN of the representative point r_ϱ. Then, by invoking Lemma A.3, we conclude that the transport plan τ is δ-close, as desired.

For any region $\varrho \in \mathcal{A}_\delta$ and any point $b \in B$ with $\tau(\varrho, b) > 0$, by δ-optimality condition (2.5),

$$y(b) - y_\delta(r_\varrho) \geq \mathrm{d}(r_\varrho, b). \tag{A.5}$$

Furthermore, for any point $b' \neq b$ in B, by δ-optimality condition (2.4),

$$y(b') - y_\delta(r_\varrho) \leq d(r_\varrho, b') + \delta. \tag{A.6}$$

Combining Equations (A.5) and (A.6),

$$d(r_\varrho, b) - y(b) \leq -y_\delta(r_\varrho) \leq d(r_\varrho, b') - y(b') + \delta,$$

or equivalently, $d_y(r_\varrho, b) \leq d_y(r_\varrho, b') + \delta$, i.e., the point b is a δ-WNN of the representative point r_ϱ. □

Next, we show that if there exists a transport plan τ from μ to ν, a set of dual weight $y(\cdot)$ for points in B, and a set of dual weights $y'(\cdot)$ for representative points of the regions in $\mathcal{A}_\delta$ that satisfy δ-optimality conditions (2.4) and (2.5) (in which $y_\delta(\cdot)$ is replaced with $y'(\cdot)$), then reassigning the dual weights based on Equation (2.3) does not violate conditions (2.4) and (2.5), i.e., the transport plan τ and dual weights $y(\cdot)$ for points in B is δ-optimal.

LEMMA A.4. *For any scale δ, if there exists a transport plan τ from μ to ν, a set of dual weights $y(\cdot)$ for points in B, and a set of dual weights $y'(\cdot)$ for representative points of regions in $\mathcal{A}_\delta$ satisfying δ-optimality conditions (2.4) and (2.5), then $\tau, y(\cdot)$ are δ-optimal.*

Proof. To prove this lemma, we show that conditions (2.4) and (2.5) hold when plugging dual weights $y(\cdot)$ for points in B and dual weights $y_\delta(\cdot)$ derived by Equation (2.3) for representative points of $\mathcal{A}_\delta$. For any region $\varrho \in \mathcal{A}_\delta$, let b_ϱ denote the weighted nearest neighbor of r_ϱ in B with respect to weights $y(\cdot)$. For any pair $(\varrho, b) \in \mathcal{A}_\delta \times B$,

$$y_\delta(r_\varrho) = y(b_\varrho) - d(a, b_\varrho) - \delta \geq y(b) - d(r_\varrho, b) - \delta;$$

therefore, the optimality condition (2.4) holds for (ϱ, b). Next, we show that the optimality condition (2.5) also holds for all pairs (ϱ, b) with $\tau(\varrho, b) > 0$. By condition (2.4) on $\tau, y(\cdot), y'(\cdot)$, for any point $b' \in B$, $y'(r_\varrho) \geq y(b') - d(r_\varrho, b') - \delta$. Therefore,

$$y(r_\varrho) \geq \max_{b' \in B}(y(b') - d(r_\varrho, b') - \delta) = y(b_\varrho) - d(r_\varrho, b_\varrho) - \delta = y_\delta(r_\varrho).$$

As a result, for the point $b \in B$ with $\tau(\varrho, b) > 0$, by condition (2.5) on $\tau, y(\cdot), y'(\cdot)$, we have

$$y(b) - y_\delta(r_\varrho) \geq y(b) - y'(r_\varrho) \geq d(r_\varrho, b),$$

and the δ-optimality condition (2.5) holds after replacing $y'(\cdot)$ with $y_\delta(\cdot)$. □

A.3 Discretizing the Continuous Distribution.

LEMMA 2.2. *For any region $\varphi \in \mathcal{A}(\mathcal{V})$, any region $\varrho \in \mathcal{A}_\delta$ inside φ, and any point $b \in B$, if $d_\delta(r_\varphi, b) \leq 4n$, then $s_\delta(\varrho, b) = d_\delta(r_\varphi, b)\delta$. Furthermore, if $d_\delta(r_\varphi, b) = 4n + 1$, then $s_\delta(\varrho, b) \geq (4n + 1)\delta$.*

Proof. For any region $\varrho \in \mathbb{A}_\delta$, suppose $b_\varrho \in B$ denotes the weighted nearest neighbor of r_ϱ with respect to weights $y(\cdot)$. For any point $b \in B$, we can rewrite the slack $s_\delta(\varrho, b)$ as follows.

$$
\begin{aligned}
s_\delta(\varrho, b) &= \left\lfloor \frac{d(r_\varrho, b) + \delta - y(b) + y_\delta(r_\varrho)}{\delta} \right\rfloor \delta \\
&= \left\lfloor \frac{d(r_\varrho, b) + \delta - y(b) + (y(b_\varrho) - d(r_\varrho, b_\varrho) - \delta)}{\delta} \right\rfloor \delta \\
&= \left\lfloor \frac{d_y(r_\varrho, b) - d_y(r_\varrho, b_\varrho)}{\delta} \right\rfloor \delta.
\end{aligned}
\tag{A.7}
$$

For each point $b \in B$, let $V_b = \mathrm{Vor}(b)$ denote the weighted Voronoi cell of the point b in the weighted Voronoi diagram $\mathrm{VD}_y(B)$. Recall that for any $i \in [1, 4n+1]$, V_b^i denotes the i-expansion of the weighted Voronoi cell of the point b. For any pair $(\varrho, b) \in \mathbb{A}_\delta \times B$, if r_ϱ lies inside V_b, then b is the WNN of r_ϱ and by Equation (A.7), $s_\delta(\varrho, b) = 0$. Otherwise, suppose the point r_ϱ lies inside V_b^i for some $i \in [1, 4n+1]$. Let $y'(\cdot)$ denote a set of dual weights for the point set B that assigns $y'(b) = y(b) + i\delta$ to the point b and $y'(b') = y(b')$ to each point $b' \neq b$ in B. Since r_ϱ lies inside V_b^i, then b is the weighted nearest neighbor of r_ϱ with respect to weights $y'(\cdot)$, i.e., $d_{y'}(r_\varrho, b) < d_{y'}(r_\varrho, b')$ for each point $b' \neq b \in B$. Therefore,

$$
\begin{aligned}
d_y(r_\varrho, b) &= d(r_\varrho, b) - y(b) = d(r_\varrho, b) - (y'(b) - i\delta) = d_{y'}(r_\varrho, b) + i\delta \\
&< d_{y'}(r_\varrho, b_\varrho) + i\delta = d_y(r_\varrho, b_\varrho) + i\delta.
\end{aligned}
$$

Plugging into Equation (A.7), $s_\delta(\varrho, b) = \left\lfloor \frac{d_y(r_\varrho, b) - d_y(r_\varrho, b_\varrho)}{\delta} \right\rfloor \delta < i\delta$ for any region $\varrho \in \mathbb{A}_\delta$ inside V_b^i. Furthermore, for any region $\varrho \in \mathbb{A}_\delta$ outside of V_b^i, the WNN of ϱ with respect to weights $y'(\cdot)$ remains to be b_ϱ and we have $d_{y'}(r_\varrho, b) > d_{y'}(r_\varrho, b_\varrho)$. Therefore,

$$
d_y(r_\varrho, b) = d_{y'}(r_\varrho, b) + i\delta > d_{y'}(r_\varrho, b_\varrho) + i\delta = d_y(r_\varrho, b_\varrho) + i\delta.
$$

Plugging into Equation (A.7), $s_\delta(\varrho, b) = \left\lfloor \frac{d_y(r_\varrho, b) - d_y(r_\varrho, b_\varrho)}{\delta} \right\rfloor \delta \geq i\delta$ for any region $\varrho \in \mathbb{A}_\delta$ outside V_b^i. Thus, for any point $b \in B$, any region $\varphi \in \mathscr{A}(\mathcal{V})$, and any $\varrho \in \mathbb{A}_\delta$ inside φ,

- if r_ϱ lies inside V_b^1, then $s_\delta(\varrho, b) = 0$. In this case, r_φ also lies inside V_b^1 and $d_\delta(r_\varphi, b) = 0$,

- if r_ϱ lies inside $V_b^{i+1} \setminus V_b^i$ for some $i \in [1, 4n]$, then $s_\delta(\varrho, b) = i\delta$. In this case, r_φ also lies in $V_b^{i+1} \setminus V_b^i$ and $d_\delta(r_\varphi, b) = i$, and

- if r_ϱ lies outside V_b^{4n+1}, then $s_\delta(\varrho, b) \geq (4n+1)\delta$. In this case, r_φ also lies outside of V_b^{4n+1} and $d_\delta(r_\varphi, b) = 4n + 1$.

This completes the proof of this lemma. $\qquad \square$

A.4 δ-Optimality of the Computed Transport Plan.

LEMMA A.5. *Let $\tau_{2\delta}, y(\cdot)$ be any 2δ-optimal transport plan from μ to ν, where the dual weights of points in B are integer multiples of 2δ. Then, for any region $\varrho \in \mathcal{A}_\delta$ and any point $b \in B$, if $\tau_{2\delta}(\varrho, b) > 0$, then $s_\delta(\varrho, b) \le 4\delta$.*

Proof. Let ϱ^* denote the region in $\mathcal{A}_{2\delta}$ containing ϱ (by construction, it can be easily confirmed that the set of valid weight vectors $\mathbb{X}_{2\delta}$ is a subset of $\mathbb{W}_\delta$ and hence, each region in $\mathcal{A}_\delta$ completely lies inside a region in $\mathcal{A}_{2\delta}$). Define b_ϱ to be the weighted nearest neighbor of r_ϱ (and consequently r_{ϱ^*}) with respect to weights $y(\cdot)$. By Equation (2.3), $y_{2\delta}(r_{\varrho^*}) = y(b_\varrho) - \mathrm{d}(r_{\varrho^*}, b_\varrho) - 2\delta$ and $y_\delta(r_\varrho) = y(b_\varrho) - \mathrm{d}(r_\varrho, b_\varrho) - \delta$. Hence,

$$
\begin{aligned}
s_\delta(\varrho, b) &= \left\lfloor \frac{\mathrm{d}(r_\varrho, b) + \delta - y(b) + y_\delta(r_\varrho)}{\delta} \right\rfloor \delta \\
&= \left\lfloor \frac{\mathrm{d}(r_\varrho, b) + \delta - y(b) + (y(b_\varrho) - \mathrm{d}(r_\varrho, b_\varrho) - \delta)}{\delta} \right\rfloor \delta \\
&= \left\lfloor \frac{\mathrm{d}_y(r_\varrho, b) - \mathrm{d}_y(r_\varrho, b_\varphi)}{\delta} \right\rfloor \delta \le \left\lfloor \frac{\mathrm{d}_y(r_{\varrho^*}, b) - \mathrm{d}_y(r_{\varrho^*}, b_\varrho)}{\delta} \right\rfloor \delta + 2\delta, \quad\quad \text{(A.8)}
\end{aligned}
$$

where the last inequality is resulted from Lemma A.6 below. Finally, from the 2δ-optimality condition (2.5) on $\tau_{2\delta}, y(\cdot)$,

$$
\mathrm{d}(r_{\varrho^*}, b) \le y(b) - y_{2\delta}(r_{\varrho^*}) = y(b) - (y(b_\varrho) - \mathrm{d}(r_{\varrho^*}, b_\varrho) - 2\delta).
$$

Hence,

$$
\mathrm{d}_y(r_{\varrho^*}, b) - \mathrm{d}_y(r_{\varrho^*}, b_\varrho) \le 2\delta. \quad\quad \text{(A.9)}
$$

Plugging Equations (A.9) into Equation (A.8),

$$
s_\delta(\varrho, b) \le \left\lfloor \frac{\mathrm{d}_y(r_{\varrho^*}, b) - \mathrm{d}_y(r_{\varrho^*}, b_\varrho)}{\delta} \right\rfloor \delta + 2\delta \le 4\delta,
$$

as claimed. $\quad\square$

LEMMA A.6. *For any region $\varrho^* \in \mathcal{A}_{2\delta}$, any pair of points $a_1, a_2 \in \varrho^*$, and any pair of points $b_1, b_2 \in B$, $\left\lfloor \frac{\mathrm{d}(a_1,b_1) - \mathrm{d}(a_1,b_2)}{\delta} \right\rfloor \delta \le \left\lfloor \frac{\mathrm{d}(a_2,b_1) - \mathrm{d}(a_2,b_2)}{\delta} \right\rfloor \delta + 2\delta$.*

Proof. To prove this lemma, we first construct a valid weight vector $w \in \mathbb{W}_{2\delta}$ such that in the Voronoi diagram $\mathrm{VD}_w(B)$, the region ϱ^* lies inside the Voronoi cell of b_1, which gives us $\mathrm{d}_w(a_1, b_1) \le \mathrm{d}_w(a_1, b_2)$. Then, we increase the weight of b_2 in w by 2δ and obtain another valid weight vector $w_+ \in \mathbb{W}_{2\delta}$ such that ϱ^* now lies inside the Voronoi cell of b_2 and conclude $\mathrm{d}_w(a_2, b_2) \le \mathrm{d}_w(a_2, b_1) + 2\delta$. Combining the two bounds, we get $\mathrm{d}(a_1, b_1) - \mathrm{d}(a_1, b_2) \le \mathrm{d}(a_2, b_1) - \mathrm{d}(a_2, b_2) + 2\delta$, leading to the lemma statement. We describe the details below.

Consider a valid weight vector $w \in \mathbb{W}_{2\delta}$ that assigns $w(b_1) = \left\lceil \frac{\mathrm{d}(r_{\varrho^*}, b_1)}{2\delta} \right\rceil 2\delta, w(b_2) = \left\lceil \frac{\mathrm{d}(r_{\varrho^*}, b_2)}{2\delta} \right\rceil 2\delta$, and $w(b') = 0$ for each $b' \ne b_1, b_2$ in B. Without loss of generality, assume

$\mathrm{d}_w(r_{\varrho^*}, b_1) < \mathrm{d}_w(r_{\varrho^*}, b_2)^5$. By construction,

$$-2\delta < \mathrm{d}_w(r_{\varrho^*}, b_1) < \mathrm{d}_w(r_{\varrho^*}, b_2) \le 0 \le \min_{b' \in B, b' \ne b_1, b_2} \mathrm{d}_w(r_{\varrho^*}, b).$$

Hence, the point r_{ϱ^*} and consequently the region ϱ^* lie inside the Voronoi cell of b_1 in $\mathrm{VD}_w(B)$. Therefore,

$$\mathrm{d}(a_1, b_1) - w(b_1) = \mathrm{d}_w(a_1, b_1) \le \mathrm{d}_w(a_1, b_2) = \mathrm{d}(a_1, b_2) - w(b_2).$$
$$\mathrm{d}(a_1, b_1) - \mathrm{d}(a_1, b_2) \le w(b_1) - w(b_2). \tag{A.10}$$

Next, consider the weight vector $w_+ \in \mathbb{W}_{2\delta}$ that assigns $w_+(b_2) = w(b_2) + 2\delta$ and $w_+(b') = w(b')$ for all points $b' \ne b_2$ in B. In this case,

$$\mathrm{d}_{w_+}(r_{\varrho^*}, b_2) \le -2\delta < \mathrm{d}_{w_+}(r_{\varrho^*}, b_1) = \mathrm{d}_w(r_{\varrho^*}, b_1) \le 0 \le \min_{b' \in B, b' \ne b_1, b_2} \mathrm{d}_w(r_{\varrho^*}, b).$$

Therefore, the point r_{ϱ^*} and consequently the region ϱ^* lie inside the Voronoi cell of b_2 in $\mathrm{VD}_{w_+}(B)$. Therefore,

$$\mathrm{d}(a_2, b_2) - (w(b_2) + 2\delta) = \mathrm{d}_{w_+}(a_2, b_2) \le \mathrm{d}_{w_+}(a_2, b_1) = \mathrm{d}(a_2, b_1) - w(b_1),$$
$$w(b_1) - w(b_2) \le \mathrm{d}(a_2, b_1) - \mathrm{d}(a_2, b_2) + 2\delta. \tag{A.11}$$

Combining Equations (A.10) and (A.11),

$$\mathrm{d}(a_1, b_1) - \mathrm{d}(a_1, b_2) \le w(b_1) - w(b_2) \le \mathrm{d}(a_2, b_1) - \mathrm{d}(a_2, b_2) + 2\delta,$$
$$\left\lfloor \frac{\mathrm{d}(a_1, b_1) - \mathrm{d}(a_1, b_2)}{\delta} \right\rfloor \delta \le \left\lfloor \frac{\mathrm{d}(a_2, b_1) - \mathrm{d}(a_2, b_2)}{\delta} \right\rfloor \delta + 2\delta.$$

□

Residual Network. Given two transport plans σ_1 and σ_2 from $\hat{\mu}_\delta$ to ν, we define the residual network $\mathscr{G}(\sigma_1, \sigma_2)$ on the vertex set $X_\delta \cup B$ as follows. Define $\sigma := \sigma_1 - \sigma_2$ to be a function that assigns, for any pair $(r, b) \in X_\delta \times B$, $\sigma(r, b) = \sigma_1(r, b) - \sigma_2(r, b)$. For any pair $(r, b) \in X_\delta \times B$, if $\sigma(r, b) > 0$, then we add an edge directed from b to r with a capacity $\sigma(r, b)$; otherwise, if $\sigma(r, b) < 0$, then we add an edge directed from r to b with a capacity $|\sigma(r, b)|$.

LEMMA A.7. *Given any two transport plans σ_1 and σ_2 from $\hat{\mu}_\delta$ to ν, for any directed edge $(r, b) \in X_\delta \times B$ in the residual network $\mathscr{G}(\sigma_1, \sigma_2)$, there exists a directed cycle C in $\mathscr{G}(\sigma_1, \sigma_2)$ that contains the edge (r, b).*

Proof. To prove this lemma, we conduct a DFS-style search from the point b in the residual network to compute a directed path P from b to r. This proves the lemma since concatenating the edge (r, b) to P results in a directed cycle on the residual network containing (r, b). Our proof relies on the following observation: Since both σ_1 and σ_2 are transport plans from $\hat{\mu}_\delta$ to ν, by the construction of the residual network, for any point

---

$u \in X_\delta \cup B$, the total capacity of incoming edges to u is equal to the total capacity of outgoing edges from u.

We conduct a DFS-style procedure that grows a path $P = \langle r = p_0, b = p_1, p_2, \ldots, p_k \rangle$ as follows. Initially, we set $P = \langle r = p_0, b = p_1 \rangle$. At each step, for the last point p_k of the path P, let $N(p_k)$ denote the set of all outgoing edges from p_k. Note that since there exists an incoming edge (p_{k-1}, p_k) in the residual graph, by the observation stated above, $N(p_k)$ is not empty. Consider any point $p \in N(p_k)$.

- If $p = r$, then $P \circ (p_k, r)$ is a cycle containing (r, b), as desired.

- Otherwise, if p already exists in the path P as p_i for some $i \geq 1$, then we have found a cycle $C = \langle p_i, p_{i+1}, \ldots, p_k, p_i \rangle$. We "cancel" this cycle as follows. Define the capacity of the cycle $c(C)$ as the minimum capacity of all edges on C. We then decrease the capacity of all edges on C by $c(C)$ and for those that now have a zero capacity, we simply remove them from the residual network. We set $P = \langle p_0 = r, p_1 = b, \ldots, p_i \rangle$ and continue our search.

- Otherwise, we add p as p_{k+1} to the path P and continue the search.

Note that since all edges in the residual network have a positive finite capacity at all times, each time we cancel a cycle reduces the total capacity of the edges of the residual network. Furthermore, the length of the path P will never be more than $2n$, as there are only n points in the set B and the residual network is a bipartite graph. Hence, our DFS-style procedure will terminate by returning a cycle containing (r, b). □

B Missing Details of Section 3

Without loss of generality, we will assume that $\varepsilon < \frac{1}{2}$. Otherwise, we can replace ε with $\min\{\frac{1}{3}, \varepsilon\}$ without increasing runtime dependence on ε. This choice of ε is important to guarantee that all neighborhoods $\mathcal{N}_\varepsilon(b)$ are disjoint.

The following lemmas roughly split the edge costs into three cases. First, we observe that we can safely transport mass to any point $b \in B$ from regions of A that have extremely small distances to b. Second, we observe that we can arbitrarily transport any mass of μ that is far enough from all points of B at the cost of an small error. Finally, given a box $\square$ with the property that for any point $b \in B$, the point b is $(1 + \varepsilon)$-approximately equidistant from all points inside $\square$, the mass of μ inside $\square$ can be moved to the center of $\square$ without too much sacrifice.

LEMMA 3.2. *Let τ^* be an optimal transport plan between μ and ν, and let $\tilde{\tau}$ be the transport plan returned by the algorithm. There exists a transport plan $\hat{\tau}$ such that (i) $\hat{\tau} = \tilde{\tau}$ when restricted to $\bigcup_{b \in B} \mathcal{N}_\varepsilon(b)$, and (ii) $\mathcal{\ell}(\hat{\tau}) \leq (1 + \varepsilon)\mathcal{\ell}(\tau^*)$.*

Proof. We break the proof of this lemma into two stages. For stage I, we argue that there exists an intermediary transport plan $\hat{\tau}_1$ between μ and ν where (i) $\int_{\mathcal{N}_\varepsilon(b)} \hat{\tau}_1(a, b) \, da$ is as large as possible for all b and (ii) $\mathcal{\ell}(\hat{\tau}_1) \leq (1 + O(\varepsilon))\mathcal{\ell}(\tau^*)$. For stage II, we argue that from $\hat{\tau}_1$, the choice of mass within each neighborhood $\mathcal{N}_\varepsilon(b)$ which is greedily coupled with b can be swapped so that $\hat{\tau}$ agrees with τ while only incurring a $(1 + \varepsilon)$ approximation error.

Stage I: Let b be an arbitrary element of B. Suppose $\int_{\mathcal{N}_\varepsilon(b)} \tau^*(x,b)\, dx <$ $\min\left\{ \int_{\mathcal{N}_\varepsilon(b)} \mu(x)\, dx, \nu(b) \right\}$, i.e. there is some mass within the approximate ball $\mathcal{N}_\varepsilon(b)$ which could be routed to b by τ^* but is instead routed to some point $b' \neq b$. Then there exist sets $U \subseteq \mathcal{N}_\varepsilon(b), V \subseteq A \setminus \mathcal{N}_\varepsilon(b)$ and $B' \subseteq B \setminus \{b\}$ where $\sum_{b' \in B'} \int_U \tau^*(u,b')\, du = \int_V \tau^*(v,b)\, dv > 0$ (some mass in $\mathcal{N}_\varepsilon(b)$ is routed away from b and an equal mass outside $\mathcal{N}_\varepsilon(b)$ is routed to b via τ^*).

Since $U \subseteq \mathcal{N}_\varepsilon(b)$, where the radius of the approximate ball $\mathcal{N}_\varepsilon(b)$ is ε and $\|b - b'\| \geq 1$ for all $b' \in B'$, we know that

$$\|u - b'\| \geq (1 - \varepsilon)\|b' - b\| \text{ and } \|u - b\| \leq \varepsilon\|b' - b\|$$

for all $u \in U$ and $b' \in B'$. Hence, $\|u - b\| \leq \frac{\varepsilon}{1-\varepsilon}\|u - b'\|$. Moreover, by the triangle inequality we can conclude that for any $u, v \in U \times V$ and $b' \in B'$,

$$\|v - b'\| \leq \|v - b\| + \|b - u\| + \|u - b'\| \leq \|v - b\| + \left(1 + \frac{\varepsilon}{1 - \varepsilon}\right)\|u - b'\|.$$

Let τ_1 be defined as the transport plan which routes the mass of U to b, the mass of V to b', and equals τ^* elsewhere. It follows that

$$\int_U \|u - b\|\hat{\tau}_1(u,b)\, du + \sum_{b' \in B'} \int_V \|v - b'\|\hat{\tau}_1(v,b')\, dv$$

$$\leq \left(1 + \frac{2\varepsilon}{1 - \varepsilon}\right) \sum_{b' \in B'} \int_U \|u - b'\|\tau^*(u,b')\, du + \int_V \|v - b\|\tau^*(v,b)\, dv.$$

Since $\varepsilon \in (0, \frac{1}{2})$, we note that $\frac{2\varepsilon}{1-\varepsilon} \leq 4\varepsilon$. Furthermore, since $\hat{\tau}_1 = \tau^*$ outside of $(U \cup V) \times B' \cup \{b\}$, we deduce $\mathcal{C}(\hat{\tau}_1) \leq (1 + 4\varepsilon)\mathcal{C}(\tau^*)$.

Finally, we note that $\mathcal{N}_\varepsilon(b) \cap \mathcal{N}_\varepsilon(b') = \varnothing$ for all $b \neq b'$ since $\varepsilon \in (0, \frac{1}{2})$. Therefore, the $(1 + 4\varepsilon)$ approximation factor is incurred at most once for each $U \subseteq \mathcal{N}_\varepsilon(b)$ which is rerouted. Since each operation reroutes a maximal amount of mass in $\mathcal{N}_\varepsilon(b)$ to b and every pair of neighborhoods is disjoint, we conclude that after n such U, V swaps a satisfactory transport plan $\hat{\tau}_1$ has been constructed from τ^*.

Stage II: Let b be an arbitrary element of B, and let $\mathcal{N}_\varepsilon(b)$ be the approximate ball of radius ε centered at b. Suppose there exist disjoint sets $X, Y \subset \mathcal{N}_\varepsilon(b)$ and some $B' \subset B \setminus \{b\}$ where

$$\int_X \tau(x,b)\, dx = \int_Y \hat{\tau}_1(y,b)\, dy = \sum_{b' \in B'} \int_X \hat{\tau}_1(x,b')\, dx > 0.$$

In the same manner as stage I, we now show that for any $x, y \in X \times Y$ and $b' \in B'$,

$$\|x - b\| + \|y - b'\| \leq (1 + 6\varepsilon)\|x - b'\| + \|y - b\|.$$

Let $x \in X, y \in Y$ and $b' \in B'$ be arbitrarily chosen. First, observe that

$$\|y - b'\| \leq \|x - b'\| + \|x - b\| + \|y - b\| \leq \|x - b'\| + 2\varepsilon$$

by triangle inequality and condition 2 of Lemma 3.1. Additionally, since $x \in X \subseteq \mathcal{N}_\varepsilon(b)$, we note that $\|x - b\| \leq \varepsilon$. Now we can use the triangle inequality to bound

$$\|x - b'\| \geq \|b - b'\| - \|x - b\| \geq 1 - \alpha.$$

Combining these inequalities and $\varepsilon \in (0, \frac{1}{2})$, we conclude

$$\|y - b'\| + \|x - b\| \leq \|x - b'\| + 3\varepsilon$$
$$\leq (1 + \frac{3\varepsilon}{1 - \varepsilon})\|x - b'\| + \|y - b\|$$
$$\leq (1 + 6\varepsilon)\|x - b'\| + \|y - b\|.$$

Let $\hat{\tau}$ be defined as the transport plan which routes the mass of Y to B', the mass of X to b, and equals $\hat{\tau}_1$ elsewhere. It follows that

$$\int_X \|x - b\|\hat{\tau}(x, b) \, dx + \sum_{b' \in B'} \int_Y \|y - b'\|\hat{\tau}(y, b') \, dy$$
$$\leq (1 + 6\varepsilon) \sum_{b' \in B'} \int_X \|x - b'\|\hat{\tau}_1(x, b') \, dx + \int_Y \|y - b\|\hat{\tau}_1(y, b) \, dv.$$

Furthermore, since $\hat{\tau}_1 = \hat{\tau}$ outside of $(X \cup Y) \times (B' \cup \{b\})$, we deduce $\cent(\hat{\tau}_1) \leq (1 + 6\varepsilon)\cent(\tau^*)$. Repeat for every $b \in B$ and every $X, Y \subseteq \mathcal{N}_\varepsilon(b)$ and by construction we then have $\hat{\tau} = \tau$ on $\mathcal{N}_\varepsilon(b) \times \{b\}$ for all $b \in B$. Note that no neighborhoods $\mathcal{N}_\varepsilon(b)$ intersect, so the cost approximation factor does not increase since no set X is swapped more than once. We conclude that $\cent(\hat{\tau}) \leq (1 + 22\varepsilon)\cent(\tau^*)$. $\square$

LEMMA 3.3. *Let $\tilde{\tau}$ be the semi-discrete transport plan constructed by our algorithm. Let τ be any arbitrary transport plan. Then,*

$$\sum_{b \in B} \int_{A \setminus \bigcup_{\square \in \mathcal{G}} \square} \|a - b\| \cdot \tilde{\tau}(a, b) \, da \leq (1 + \varepsilon) \sum_{b \in B} \int_{A \setminus \bigcup_{\square \in \mathcal{G}} \square} \|a - b\| \cdot \tau(a, b) \, da.$$

Proof. Suppose $a \in A \setminus \bigcup_{\square \in \mathcal{G}} \square$. Then a satisfies $\|a - b_1\| \geq \frac{1}{\varepsilon}\|b_1 - b_2\|$ for all $b_1, b_2 \in B$. By the triangle inequality, we note that $\|a - b_2\| \leq (1 + \varepsilon)\|a - b_1\|$ for all such $a \in A$ and $b_1, b_2 \in B$.

Since τ and $\tilde{\tau}$ are both feasible transport plans, they satisfy $\sum_{b \in B} \tau(a, b) = \sum_{b \in B} \tilde{\tau}(a, b)$. We deduce that

$$\sum_{b \in B} \|a - b\| \cdot \tau(a, b) \leq (1 + \varepsilon) \sum_{b \in B} \|a - b\| \cdot \tilde{\tau}(a, b)$$

by combining the previous two statements. Finally, integrating over all $a \in A \setminus \bigcup_{\square \in \mathcal{G}} \square$ gives us the desired result

$$\sum_{b \in B} \int_{A \setminus \bigcup_{\square \in \mathcal{G}} \square} \|a - b\| \cdot \tau(a, b) \, da \leq (1 + \varepsilon) \sum_{b \in B} \int_{A \setminus \bigcup_{\square \in \mathcal{G}} \square} \|a - b\| \cdot \tilde{\tau}(a, b) \, da.$$

$\square$

LEMMA 3.4. *Let $\hat{\tau}'$ be a transport plan between μ' and ν' defined by $\hat{\tau}'(a, b) = \hat{\tau}(a, b)$ if $a \notin \mathcal{N}_\varepsilon(b)$ and $\hat{\tau}'(a, b) = 0$ otherwise. Then $\cent(\tau_\sigma) \leq (1 + \varepsilon)\cent(\hat{\tau}')$.*

Proof. Note that

$$\mathcal{c}(\tau) = \sum_{b \in B} \int_{A \backslash \mathcal{N}_\varepsilon(b)} \|a - b\| \tau(a, b) \, da + \sum_{b \in B} \int_{\mathcal{N}_\varepsilon(b)} \|a - b\| \tau(a, b) \, da$$

where $\mathcal{N}_\varepsilon(b)$ again denotes the approximate ball centered at b of radius ε. We can analogously claim

$$\mathcal{c}(\hat{\tau}) = \sum_{b \in B} \int_{A \backslash \mathcal{N}_\varepsilon(b)} \|a - b\| \hat{\tau}(a, b) \, da + \sum_{b \in B} \int_{\mathcal{N}_\varepsilon(b)} \|a - b\| \hat{\tau}(a, b) \, da$$

$$= \sum_{b \in B} \int_{A \backslash \mathcal{N}_\varepsilon(b)} \|a - b\| \hat{\tau}(a, b) \, da + \sum_{b \in B} \int_{\mathcal{N}_\varepsilon(b)} \|a - b\| \tau(a, b) \, da,$$

where the second equality follows from the fact that $\hat{\tau} = \tau$ on $\cup_{b \in B}(\mathcal{N}_\varepsilon(b) \times b)$. It therefore suffices to compare the transport plans on the pairs $a, b \in A \times B$ of (approximate) distance greater than ε. That is,

$$\mathcal{c}(\tau) - \mathcal{c}(\hat{\tau}) = \sum_{b \in B} \int_{A \backslash \mathcal{N}_\varepsilon(b)} \|a - b\| (\tau(a, b) - \hat{\tau}(a, b)) \, da.$$

For simplicity, let $\mathscr{Z} = \cup_{\square \in \mathcal{G}} \square$, $\mathscr{X} = A \backslash \mathscr{Z}$, and define $\mathscr{Z}_b := \mathscr{Z} \backslash \mathcal{N}_\varepsilon(b)$ for each $b \in B$. Then, we observe

$$\mathcal{c}(\tau) - \mathcal{c}(\hat{\tau}) = \sum_{b \in B} \left[\int_{\mathscr{X}} \|a - b\| (\tau(a, b) - \hat{\tau}(a, b)) \, da + \int_{\mathscr{Z}_b} \|a - b\| (\tau(a, b) - \hat{\tau}(a, b)) \, da \right].$$

For convenience, define $\tau' = \tau - \hat{\tau}$ and let $\mathcal{G}_b = \{\square \in \mathcal{G} : \square \subseteq \mathcal{N}_\varepsilon(b)\}$. Additionally define the discrete plans $\hat{\sigma}$ and σ' by $\hat{\sigma}(c_\square, b) := \int_\square \hat{\tau}(a, b) \, da$ and $\sigma'(c_\square, b) = \sigma(c_\square, b) - \hat{\sigma}(c_\square, b)$. We conclude that

$$\mathcal{c}(\tau) - \mathcal{c}(\hat{\tau}) = \sum_{b \in B} \left[\int_{\mathscr{X}} \|a - b\| \tau'(a, b) \, da + \int_{\mathscr{Z}_b} \|a - b\| \tau'(a, b) \, da \right]$$

$$\leq \sum_{b \in B} \left[\int_{\mathscr{X}} \|a - b\| \tau'(a, b) \, da + (1 + \varepsilon) \sum_{\square \in \mathcal{G} \backslash \mathcal{G}_b} \|c_\square - b\| \sigma'(c_\square, b) \right]$$

$$\leq \varepsilon \sum_{b \in B} \left[\int_{\mathscr{X}} \|a - b\| \hat{\tau}(a, b) \, da + (1 + \varepsilon) \sum_{\square \in \mathcal{G} \backslash \mathcal{G}_b} \|c_\square - b\| \hat{\sigma}(c_\square, b) \right]$$

$$\leq \varepsilon \sum_{b \in B} \left[\int_{\mathscr{X}} \|a - b\| \hat{\tau}(a, b) \, da + (1 + \varepsilon)^2 \int_{\mathscr{Z}_b} \|a - b\| \hat{\tau}(a, b) \, da \right]$$

$$\leq \frac{9}{4} \varepsilon \sum_{b \in B} \left[\int_{\mathscr{X}} \|a - b\| \hat{\tau}(a, b) \, da + \int_{\mathscr{Z}_b} \|a - b\| \hat{\tau}(a, b) \, da \right],$$

where the second and fourth lines follow from the first condition of Lemma 3.1, the third line follows from Lemma 3.3 and the fact that σ is a $(1 + \varepsilon)$-approximate transport plan, and the last line uses $\varepsilon \in (0, \frac{1}{2})$. We conclude that $\mathcal{c}(\tau) \leq (1 + \frac{9}{4} \varepsilon) \mathcal{c}(\hat{\tau})$. $\square$

C Missing Proofs of Section 4

LEMMA 4.1. *The graph $\mathcal{G}$ contains $O(nh)$ vertices and $O(n\varepsilon^{-d}h)$ edges. The max degree of any vertex in $\mathcal{G}$ is at most $O(\varepsilon^{-d}\log n)$. Furthermore, for any pair of points (a,b), $\phi(P_{a,b}) \geq \|a - b\|$ and $\mathbb{E}\left[\phi(P_{a,b})\right] \leq (1 + 3\varepsilon)\|a - b\|$.*

Proof. The vertex set of our graph consists of the center points of all non-empty cells of the quad-tree as well as the point sets $A \cup B$. At each level i of the tree, the total number of non-empty cells of level i is no more than n. Since our spanner contains the center point of each non-empty cell at all levels, where $h = O(\log \log n)$, the total number of vertices is $O(n \log \log n)$.

Next, we bound the number of edges of our graph. For any cell $\square$, we add two sets of edges corresponding to two $(1 + \varepsilon)$-spanners $\mathcal{S}_\square$ and $\mathcal{S}'_\square$. Each one of these spanners has $O(n_\square \varepsilon^{-d})$ edges and bounded degree of $O(n_\square \varepsilon^{-d} \log n) \leq O(n \varepsilon^{-d} \log n)$ for cell $\square$. In each level of the graph, there are at most n points distributed among cells where each point appears at most once. Therefore, $|E|$ is bounded by a sum over all levels ℓ of the graph:

$$O(\sum_\square n_\square \varepsilon^{-d}) \leq O(\sum_\ell n\varepsilon^{-d}) = O(n\varepsilon^{-d}h).$$

The cost of any edge in the spanner $\mathcal{G}$ is the Euclidean distance of the two endpoints of the edge. Therefore, from the triangle inequality, any path from a to b has a cost of at least the Euclidean distance of a and b; i.e, $\phi(P_{a,b}) \geq \|a - b\|$.

Suppose (a,b) have least common ancestor $\square$. Let ξ_a, ξ_b be the subcells in $\mathcal{S}'_\square$ which contain a and b, respectively. Since $\mathcal{S}'_\square$ is a $(1 + \varepsilon)$-spanner, the length of the shortest path from ξ_a to ξ_b is a $(1 + \varepsilon)$-approximation of their Euclidean distance.

Define P_a and P_b to be the shortest paths from a to ξ_a and b to ξ_b, respectively, only taking greedy edges. Then, one path from a to b in the graph is the following path:

$$P = P_a \circ P_{a,b}^\square \circ P_b.$$

For any cell $\square$, define $\delta_\square = \sqrt{d}\varepsilon h^{-1}\ell_\square$ to be the diameter of the subcells of $\square$. Define δ_a and δ_b to be the diameter of the subcells ξ_a and ξ_b. Recall that $\square$ is of level $i > 0$.

For any $\square' \in C[\square]$, we note that the shortest path from $\square'$ to $\square$ is bounded above in length by $(1 + \varepsilon)\|c_{\square'} - c_\square\|$. Using this, we can bound the length of P_a and P_b by the greedy paths going directly up the tree:

$$\phi(P_a) + \phi(P_b) \leq \frac{1}{2}(\delta_a + \delta_b).$$

Next, we bound the expected value of δ_a and δ_b. For any level j of the tree, the probability that the least common ancestor of (a, b) is of level j is

$$\Pr\left[\text{lev}(a, b) = j\right] \leq \frac{\sqrt{d}\|a - b\|}{\ell_{j+1}}.$$

As a result,

$$\mathbb{E}\left[\delta_a\right] \leq \sum_{j=1}^{h} \Pr\left[\text{lev}(a, b) = j\right].\delta_{j+1} \leq \sum_{j=1}^{h} \frac{\sqrt{d}\|a - b\|}{\ell_{j+1}} \cdot \frac{\varepsilon\ell_{j+1}}{2\sqrt{dh}} = \frac{\varepsilon}{2}\|a - b\|.$$

An analogous claim can be made for δ_b. Finally, as discussed before, the cost of the shortest path between c_{ξ_a}, c_{ξ_b} is bounded above by $(1+\varepsilon)\|c_{\xi_a} - c_{\xi_b}\|$. Using triangle inequality,

$$\|c_{\xi_a} - c_{\xi_b}\| \leq \|a - b\| + \frac{1}{2}(\delta_a + \delta_b)$$

Combining all these bounds,

$$\mathbb{E}\left[\phi(P_{a,b})\right] \leq \mathbb{E}\left[\phi(P)\right] = \mathbb{E}\left[\phi(P_a) + \phi(P_{ab}^{\square}) + \phi(P_b)\right]$$

$$\leq \mathbb{E}\left[(1+\varepsilon)\|c_{\xi_a} - c_{\xi_b}\| + \frac{1}{2}(\delta_a + \delta_b)\right]$$

$$\leq (1+\varepsilon)\|a - b\| + \frac{1}{2}((1+\varepsilon)+1)\mathbb{E}\left[\delta_a + \delta_b\right]$$

$$\leq \left((1+\varepsilon) + \frac{(2+\varepsilon)\varepsilon}{2}\right)\|a - b\| \leq (1+\frac{5}{2}\varepsilon)\|a - b\|.$$

where the last inequality assumes $\varepsilon \leq 1$. If not, then ε can be substituted for 1 without loss of generality. To obtain $(1+\varepsilon)$-approximation instead, one can rescale ε by $\frac{2}{5}$.

$\square$

LEMMA 4.2. *For any edge $(u,v) \in E$, $|y(u) - y(v)| \leq O(d^{3/2}h\varepsilon^{-1})\|u - v\|$.*

Proof. For any edge $(u,v) \in E$, consider the following cases.

1. *Greedy edges:* If (u,v) is an greedy edge, by the definition, there exists a cell $\square$ such that $u, v \in \mathscr{I}_{\square}$. Let $(f_{\square}, y_{\square})$ denote the flow and the set of dual weights computed on the local instance $\mathscr{I}_{\square}$. From the properties of exact primal-dual minimum cost flow, $|y_{\square}(u) - y_{\square}(v)| \leq \|u - v\|$. Therefore, by the dual assignment of our algorithm,

$$|y(u) - y(v)| = |(y_{\square}(u) - y_{\square}(c_{\square}) + y(c_{\square})) - (y_{\square}(v) - y_{\square}(c_{\square}) + y(c_{\square}))| \leq \|u - v\|.$$

2. *Shortcut edges:* If (u,v) is a shortcut edge, then there exists a cell $\square$ of level i and children $\square_1, \square_2 \in \mathsf{C}[\square]$ such that u (resp, v) is the center point of a subcell $\xi_1 \in \mathsf{S}[\square_1]$ (resp. $\xi_2 \in \mathsf{S}[\square_2]$); i.e, $u = c_{\xi_1}$ (resp. $v = c_{\xi_2}$). Observe that $c_{\xi_1} \in \mathscr{I}_{\square_1}$ and $c_{\xi_2} \in \mathscr{I}_{\square_2}$. Recall that $\mathscr{S}_{\square_1}$ (resp. $\mathscr{S}_{\square_2}$) denotes the $(1+\varepsilon)$-spanner constructed on the local instance $\mathscr{I}_{\square_1}$ (resp. $\mathscr{I}_{\square_2}$). Let $P = \langle c_{\xi_1} = p_1, \ldots, p_{k_1} = c_{\square_1}\rangle$ be the path in $\mathscr{S}_{\square_1}$ from c_{ξ_1} to $c_{\square_1}$. Similarly, let $Q = \langle c_{\xi_2} = q_1, \ldots, q_{k_2} = c_{\square_2}\rangle$ be the path in $\mathscr{S}_{\square_2}$ connecting c_{ξ_1} to $c_{\square_1}$. Finally, note that $c_{\square_1}, c_{\square_2} \in \mathscr{I}_{\square}$ and let $R = \langle c_{\square_1} = r_1, \ldots, r_{k_3} = c_{\square_2}\rangle$ be the path connecting the two center points $c_{\square_1}$ and $c_{\square_2}$ in $\mathscr{S}_{\square}$. All the edges in the paths P, Q, and R are greedy edges. By the triangle inequality,

$$|y(c_{\xi_1}) - y(c_{\xi_2})| \leq |y(c_{\xi_1}) - y(c_{\square_1})| + |y(c_{\square_1}) - y(c_{\square_2})| + |y(c_{\square_2}) - y(c_{\xi_2})|$$

$$\leq \left(\sum_{j=1}^{k_1-1} |y(p_j) - y(p_{j+1})|\right) + \left(\sum_{j=1}^{k_3-1} |y(r_i) - y(r_{i+1})|\right)$$

$$+ \left(\sum_{j=1}^{k_2-1} |y(q_{j+1}) - y(q_j)|\right)$$

$$\leq (1+\varepsilon)\|c_{\xi_1} - c_{\square_1}\| + (1+\varepsilon)\|c_{\square_1} - c_{\square_2}\| + (1+\varepsilon)\|c_{\square_2} - c_{\xi_2}\|.$$

Since ξ_1 and ξ_2 are subcells of children $\square_1$ and $\square_2$ of $\square$, their side-lengths are both $\frac{\varepsilon\ell_{\square_1}}{2dh}$. Thus, the Euclidean distance of their centers is $\|c_{\xi_1} - c_{\xi_2}\| \geq \frac{\varepsilon\ell_{\square_1}}{2dh}$. Furthermore, $\|c_{\xi_1} - c_{\square_1}\| \leq \sqrt{d}\ell_{\square_1}$ and $\|c_{\square_2} - c_{\xi_2}\| \leq \sqrt{d}\ell_{\square_2}$. Combining these inequalities gives

$$\|c_{\xi_1} - c_{\square_1}\| \leq \sqrt{d}\ell_{\square_1} \leq \frac{2d^{3/2}h}{\varepsilon}\|c_{\xi_1} - c_{\xi_2}\|,$$

and the analogous for $\|c_{\square_2} - c_{\xi_2}\|$. By triangle inequality, we can extend this to conclude $\|c_{\square_1} - c_{\square_2}\| \leq O(\frac{d^{3/2}h}{\varepsilon})\|c_{\xi_1} - c_{\xi_2}\|$. Therefore,

$$|y(c_{\xi_1}) - y(c_{\xi_2})| \leq (1+\varepsilon)\|c_{\xi_1} - c_{\square_1}\| + (1+\varepsilon)\|c_{\square_1} - c_{\square_2}\| + (1+\varepsilon)\|c_{\square_2} - c_{\xi_2}\|$$
$$\leq O\left(\frac{d^{3/2}h}{\varepsilon}\right)\|c_{\xi_1} - c_{\xi_2}\|.$$

$\square$

LEMMA 4.3. $\sum_{(u,v)\in E} \sigma(u,v)\|u - v\| \leq \sum_{u\in V} y(u)\eta(u)$.

Proof. By construction, for any shortcut edge $(u,v) \in E$, $\sigma(u,v) > 0$. For any greedy edge $(u,v) \in E$, there exists a unique cell $\square$ such that $u,v \in \mathscr{I}_\square$ and the spanner $\mathscr{S}_\square$ contains the edge (u,v). By the dual assignment, if $\sigma_\square(u,v) > 0$, then

$$y(u) - y(v) = y_\square(u) - y_\square(v) = \|u - v\|.$$

Therefore, for any edge (u,v) carrying a positive flow in σ, $y_\square(u) - y_\square(v) = \|u - v\|$. As a result,

$$\sum_{(u,v)\in E} \sigma(u,v)\|u - v\| = \sum_{(u,v)\in E} \sigma(u,v)(y(u) - y(v))$$
$$= \sum_{w\in V}\left(\sum_{z:(w,z)\in E} \sigma(w,z)\right) y(w)$$
$$= \sum_{w\in V} y(w) \cdot \eta(w).$$

$\square$

D   The Multiplicative Weight Update Framework

At a very high level, the multiplicative weights method uses an approximate oracle to estimate the best flow and iteratively updates the flow using the oracle as a rough guide. In our setting, we construct an undirected graph $\mathscr{G} = (V, E)$ with $A \cup B \subseteq V$ that is a (randomized) spanner of $A \cup B$ under Euclidean distance and we compute an ε-approximate MCF on $\mathscr{G}$. The approximate oracle is a greedy algorithm GREEDY which routes flow along tree edges. This greedy tree flow leads to high costs for some pairs which have positive flow, and the multiplicative weights method gradually reroutes the flow along shorter paths between these two points in the graph.

We use complementary slackness to guide which edges are valuable. Using the LP duality, the MCF problem can be formulated as computing dual weights y maximizing $\sum_{v \in V} \eta(u)y(u)$ subject to $y(u) - y(v) \leq \|u - v\|$ for all $(u, v) \in E$. Equivalently, the collection of constraints can be expressed as $\max_{(u,v) \in E} \frac{y(u) - y(v)}{\|u - v\|} \leq 1$. We refer to the expression $\frac{y(u) - y(v)}{\|u - v\|}$ as the *slack* of an edge (u, v). By complementary slackness, if (σ, y) is an optimal primal-dual pair, then $\sigma(u, v)$ is positive when the slack of (u, v) is 1. In view of this observation, if the slack of a directed edge e is large, the MWU method increases the flow along e.

We transform the undirected graph $\mathcal{G}$ into a directed graph $G = (V, E)$ that takes the vertices of $\mathcal{G}$ and adds both directed edges for every undirected edge of $\mathcal{G}$. Additionally set $\eta(u)$ to be the original demand of $u \in A \cup B$ and 0 for $u \in V \setminus (A \cup B)$. The cost of an edge $e = (u, v)$ in G, denoted by $|e|$, is $\|u - v\|$. Given G, a demand function $\eta \colon V \to \mathbb{R}$, and a parameter $\varepsilon > 0$, the MWU algorithm computes a flow function $\sigma \colon E \to \mathbb{R}_{\geq 0}$ that satisfies the demand and $\mathcal{C}(\sigma) \leq (1 + \varepsilon)w^*$, where w^* is the min-cost flow for (G, η). The algorithm assumes the existence of a greedy algorithm $\text{GREEDY}(G, \eta)$ that computes a primal-dual pair (σ, y) on G, where $\sigma \colon E \to \mathbb{R}$ is a flow function that routes the demand η, i.e. $\sum_{v:(u,v) \in E} \sigma(u, v) - \sigma(v, u) = \eta(u)$ for all $u \in V$, and $y \colon V \to \mathbb{R}$ is a dual weight function that satisfies the following two conditions:

(C1) $|y(u) - y(v)| \leq \rho \|u - v\| \quad \forall (u, v) \in E$,

(C2) $\sum_{(u,v) \in E} \sigma(u, v) \|u - v\| \leq \sum_{u \in V} y(u)\eta(u)$,

where $\rho > 0$ is a parameter. (C1) guarantees ρ-approximate feasibility of the computed dual weights. (C2) is a strong-duality condition used to prevent GREEDY from returning trivial dual weights, as well as upper bound the cost of the flow σ.

We now describe the algorithm in more detail. Using one of the known algorithms [15, 27], we first compute an estimate of the OT cost within a $d \log n$ factor in $O(n \log n)$ time, i.e. it returns a value $\tilde{g}$ such that $w^* \leq \tilde{g} \leq (d \log n) \cdot w^*$. We refer to this algorithm as LOGAPPROX. Using this estimate, we perform an exponential search in the range $\left[\frac{\tilde{g}}{d \log n}, \tilde{g}\right]$ with increments of factor $(1 + \varepsilon)$. For any guess value g, the MWU algorithm either returns a flow $\sigma \colon E \to \mathbb{R}$ with $\mathcal{C}(\sigma) \leq (1 + \varepsilon)g$ or returns dual weights as a certificate that $g < w^*$. We now describe the MWU algorithm for a fixed value of g.

Set $T = 4\rho^2 \varepsilon^{-2} \log |E|$. The algorithm runs in at most T iterations, where in each iteration, it maintains a pre-flow vector σ^t such that $\mathcal{C}(\sigma^t) \leq g$. The flow σ^t need not route all demand successfully. Initially, set $\sigma^0(e) = \frac{g}{|e| \cdot |E|}$ so that $\mathcal{C}(\sigma^0) \leq g$. For each iteration t, define the *residual demand* of iteration t, denoted by η^t_{res}, as

$$\eta^t_{\text{res}}(u) = \eta(u) - \sum_{v:(u,v) \in E} (\sigma^{t-1}(u, v) - \sigma^{t-1}(v, u)).$$

Let $(\sigma^t_{\text{res}}, y^t)$ be the primal-dual flow computed by the GREEDY for the residual demands η^t_{res}. Recall that $(\sigma^t_{\text{res}}, y^t)$ satisfies (C1) and (C2). If $\langle \eta^t_{\text{res}}, y^t \rangle \leq \varepsilon g$, then (C2) implies that $\mathcal{C}(\sigma^t_{\text{res}}) \leq \varepsilon g$. Since σ^t_{res} routes the residual demands, the flow function $\sigma^t = \sigma^{t-1} + \sigma^t_{\text{res}}$

routes the original demand η with a cost $\not{c}(\sigma^t) \leq (1+\varepsilon)g$. In this case, the algorithm returns σ^t as the desired flow and terminates.

Otherwise, $\langle \eta_{\text{res}}^t, y^t \rangle > \varepsilon g$ and we update the flow along each edge $e = (u,v)$ of G based on the slack $s^t(u,v) = \frac{y^t(u) - y^t(v)}{\|u-v\|}$ of e with respect to dual weights y^t:

$$\sigma^t(u,v) \leftarrow \exp\left(\frac{\varepsilon}{2\rho^2} s^t(u,v)\right) \cdot \sigma^{t-1}(u,v).$$

We emphasize that flow along an edge is increasing if the slack is large. Then, one needs to rescale σ^t so that its cost is bounded above by g. If the algorithm does not terminate within T rounds, we conclude that the value of g is smaller than the MCF cost. We increase g by a factor of $(1+\varepsilon)$ and repeat the MWU algorithm.

Algorithm 1 Minimum-Cost-Flow(G, η, ε):

$\tilde{g} \leftarrow \text{LOGAPPROX}(G, \eta)$
$g \leftarrow \frac{\tilde{g}}{d \log n}, T = 4\rho^2 \varepsilon^{-2} \log |E|$
repeat
    $\sigma^0(u,v) = \sigma^0(v,u) = \frac{g}{\|u-v\| \cdot |E|} \quad \forall (u,v) \in E$
    for $t = 1, \dots, T$ **do**
        $\eta_{\text{res}}^t(u) \leftarrow \eta(u) - \left(\sum_{v:(u,v)\in E} \left[\sigma^{t-1}(u,v) - \sigma^{t-1}(v,u)\right]\right) \quad \forall u \in V$
        $(\sigma_{\text{res}}^t, y^t) \leftarrow \text{GREEDY}(G, \eta_{\text{res}}^t)$
        if $\langle \eta_{\text{res}}^t, y^t \rangle \leq \varepsilon \cdot g$ **then**
            return $\sigma^{t-1} + \sigma_{\text{res}}^t$
        end if
        $s^t(u,v) \leftarrow \frac{y^t(u) - y^t(v)}{\|u-v\|} \quad \forall (u,v) \in E$
        $\sigma^t(u,v) \leftarrow \exp(\beta s^t(u,v)) \cdot \sigma^{t-1}(u,v)$
        Rescale σ^t so $\not{c}(\sigma^t) \leq g$
    end for
    $g \leftarrow (1+\varepsilon)g$
until flow is returned

The following Lemma regarding Algorithm 1 is proven in [50], which we follow closely in this work for its use of primal-dual oracles.

LEMMA D.1. *Given an $O(d \log n)$ approximate guess g of the minimum cost flow value and an algorithm* GREEDY *which computes a primal-dual pair satisfying conditions (C1) and (C2) in $T_\rho(n)$ time, a $(1+\varepsilon)$-approximate minimum cost flow on G can be computed in $O((T_\rho(n) + |E|)\frac{\rho^2 \log n}{\varepsilon^2} \log(d \log n))$ time.*

D.1 Recovering a Transport Map To be precise, the multiplicative weights algorithm we have described so far produces a min-cost flow on the $(1+\varepsilon)$-spanner in expectation. A true transportation map is over $A \times B$. We briefly describe the procedure of [29] for completeness, which takes a flow on some approximate spanner with bounded degree and produces a transportation map. The basic idea is to iteratively skip over any vertex which has flow passing through. Algorithm 2 shows how to shortcut vertices.

Algorithm 2 Recover-Transport-Map(G, A, B, τ):

Let $f(u, v) = \tau(u, v) - \tau(v, u)$
while there exists $v \in V \setminus (A \cup B)$ where $f(u, v), f(v, w) > 0$ for some u, w **do**
    Add (u, w) to G if $(u, w) \notin E$
    $f(u, w) \leftarrow \min\{f(u, v), f(v, w)\}$
    Subtract $\min\{f(u, v), f(v, w)\}$ from both $f(u, v)$ and $f(v, w)$
    Remove (u, v) from G if $f(u, v) = 0$ (analogous for (v, w))
    if $\deg(u) > \deg(v)$ and $u \notin A \cup B$ (analogous for w) **then**
        Shortcut u (or w) in next iteration of while loop
    end if
end while

LEMMA D.2. *Given a graph $G = (V, E)$ with $A \cup B \subseteq V$ and maximum degree of $\deg_{max}$, as well as a flow τ over G which routes the demand of $A \cup B$, Algorithm 2 returns a transportation plan over $A \times B$ in $O(|E| \cdot \deg_{max})$ time.*