Learning Ultrametric Trees for Optimal Transport Regression

Samantha Chen[1], Puoya Tabaghi[2], Yusu Wang[2]

[1]Department of Computer Science and Engineering, University of California, San Diego
[2] Halıcıoğlu Data Science Institute, University of California, San Diego
sac003@ucsd.edu, ptabaghi@ucsd.edu, yusuwang@ucsd.edu

Abstract

Optimal transport provides a metric which quantifies the dissimilarity between probability measures. For measures supported in discrete metric spaces, finding the optimal transport distance has cubic time complexity in the size of the space. However, measures supported on trees admit a closed-form optimal transport that can be computed in linear time. In this paper, we aim to find an optimal tree structure for a given discrete metric space so that the tree-Wasserstein distance approximates the optimal transport distance in the original space. One of our key ideas is to cast the problem in ultrametric spaces. This helps us optimize over the space of ultrametric trees — a mixed-discrete and continuous optimization problem — via projected gradient decent over the space of ultrametric matrices. During optimization, we project the parameters to the ultrametric space via a hierarchical minimum spanning tree algorithm, equivalent to the closest projection to ultrametrics under the supremum norm. Experimental results on real datasets show that our approach outperforms previous approaches (e.g. Flowtree, Quadtree) in approximating optimal transport distances. Finally, experiments on synthetic data generated on ground truth trees show that our algorithm can accurately uncover the underlying trees.

1 Introduction

First formulated by Gaspard Monge in 18th-century France, the optimal transport problem is often explained by analogy to the problem of minimizing the time spent transporting coal from mines to factories. More formally, given two distributions and a transportation cost, optimal transport aims to find the lowest-cost way of moving points from the first distribution to the second one. When the transportation cost is a metric, the optimal transport distance is also called the 1-Wasserstein distance (Villani 2009).

Optimal transport is applied in many areas such as machine learning (Solomon et al. 2014; Frogner et al. 2015; Montavon, Müller, and Cuturi 2016; Kolouri et al. 2017; Arjovsky, Chintala, and Bottou 2017; Genevay et al. 2016; Lee and Raginsky 2018), statistics (El Moselhy and Marzouk 2012; Reich 2013; Panaretos and Zemel 2016), and computer graphics (Dominitz and Tannenbaum 2009; Rubner, Tomasi, and Guibas 2000; Rabin et al. 2011; Lavenant et al. 2018; Solomon et al. 2015). Since the optimal transport distance is computationally expensive (cubic in the number of points), several methods have emerged to efficiently approximate optimal transport distance. One of the most popular methods is the Sinkhorn distance, which uses entropic regularization to compute an approximation of optimal transport in *quadratic* time (Cuturi 2013).

Another approach relies on approximating the original metric with a tree metric (Evans and Matsen 2012; Le et al. 2019a; Indyk and Thaper 2003; Takezawa et al. 2021; Yamada et al. 2022). While this approach yields a coarser approximation of the optimal transport, it has *linear* time complexity with respect to the size of the metric space. The classic Quadtree is one of the most widely used tree approximations methods. It recursively partitions a metric space into four quadrants to construct internal nodes and represents each element of the original discrete metric space as a leaf node. Quadtree Wasserstein distance approximates the true optimal transport distance with a logarithmic distortion (Indyk and Thaper 2003). Flowtree (Backurs et al. 2019) and sliced-tree Wasserstein (Le et al. 2019b) are Quadtree-based methods designed to improve over the 1-Wasserstein distance approximation of the Quadtree algorithm. A main drawback of Quadtree-based methods is that they require a Euclidean embedding of the original discrete metric space. In contrast, clustertree is a tree-based approximation that does not require a Euclidean embedding of the original discrete metric space (Le et al. 2019b). Given a *fixed* Quadtree or clustertree topology, Yamada et al. (2022) propose a state-of-the-art method based on *optimizing the weights* on the tree to best approximate Wasserstein distances but does not change the topology of the input tree.

Our goal is to find the tree *topology and weight* that closely approximates the Wasserstein distance in the original metric space. To achieve this, we introduce a projected gradient descent procedure over the space of ultrametrics to find a tree that approximates the original Wasserstein distances. As constraining the problem to tree metrics is challenging in general, we instead use ultrametrics — a subfamily of tree metrics — as a proxy for tree weights and structure.

Problem Statement (*informal*) Let $\mathcal{X}$ be a point set in a metric space and $\mathrm{W}_1(\cdot,\cdot)$ be the optimal transport distance. We want to learn an ultrametric on $\mathcal{X}$ such that ultrametric optimal transport W_u approximates W_1. To achieve this, we cast this as a regression problem over ultrametric trees.

The key advantage of optimizing via ultrametrics is that we can project any (semi)metric to the ultrametrics; see Section 3. This projection allows us to optimize in the space of ultrametrics via a projected gradient descent-type procedure. Our contributions are as follows.

1. We define a quadratic cost function to measure the discrepancy between the true and ultrametric optimal transport distances. This cost does not require the point positions a priori but rather is parameterized by pairwise dissimilarity measures between points, i.e., it *does not* assume that input points are embedded, or even equipped with a metric.

2. We then propose a projected gradient descent method to perform optimization in ultrametric spaces. The proposed optimization process learns a weighted tree structure. Our proposed method adjusts *both tree weights and structure* throughout training. This is a novel contribution to existing work on tree-Wasserstein approximation. In previous methods, either the tree structure is fixed (Yamada et al. 2022; Backurs et al. 2019) or it is determined by approximating the discrete metric — not the Wasserstein distances (Indyk and Thaper 2003; Le et al. 2019b).

3. The learned ultrametric trees provide more accurate optimal transport approximations compared to Flowtree and Quadtree for various real-world datasets. Our method performs slightly worse than the weight-optimized methods (Yamada et al. 2022) on real-world distributions *with sparse support*; however, for denser synthetic distributions, it outperforms all aforementioned methods. The computational complexity of approximating the optimal transport distance at inference time with our learned ultrametric tree is $O(N)$ similar to other tree approximation methods.

Notation. For $N \in \mathbb{N}$, we define $[N] = \{1, \dots, N\}$. We denote the set of nonnegative real numbers as $\mathbb{R}_+$. For a vector $x \in \mathbb{R}^d$, we denote its ℓ_p norm as $\|x\|_p$. Given a metric space $\mathcal{X}$, we denote the space of probability measures over $\mathcal{X}$ as $\mathcal{P}(\mathcal{X})$. For any $x, y \in \mathbb{R}$, we let $x \wedge y = \max\{x, y\}$. For matrices $X, Y \in \mathbb{R}^{d_1 \times d_2}$, we let $\langle X, Y \rangle = \operatorname{tr}(X^\top Y)$. Let $\mathcal{X}$ to be a finite discrete set. A semimetric over $\mathcal{X}$ is the function $d_s : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ that does not necessarily satisfy the triangle inequality.

2 Preliminaries

Wasserstein Distance. The Wasserstein distance provides a metric for the space of probability distributions supported on a compact metric space. We focus on the 1-Wasserstein distance (or the optimal transport distance) for discrete probability distributions. For a discrete set $\mathcal{X} = \{x_n : n \in [N]\}$, we can compute the Wasserstein distance by solving the following linear programming problem:

$$\mathrm{W}_1(\mu, \rho) = \min_{\Pi \in \mathbb{R}_+^{N \times N}} \{\langle \Pi, D \rangle : \Pi 1 = \rho, \Pi^\top 1 = \mu\} \quad (1)$$

where $D = \big(d(x_{n_1}, x_{n_2})\big)_{n_1, n_2 \in [N]} \in \mathbb{R}_+^{N \times N}$ is the distance matrix, measures μ and ρ are N-dimensional vectors, viz., $1^\top \mu = 1, \mu \geq 0$. When D is any arbitrary cost matrix, we can still solve the optimal transport problem. Solving this linear programming problem has a time complexity of $O(N^3 \log N)$ (Pele and Werman 2009).

Tree Wasserstein Distance. Consider a weighted tree $T = (V, E)$ with metric $d_T \in V \times V \to \mathbb{R}_+$. For nodes $v_1, v_2 \in V$, let P_{v_1, v_2} be the unique path between them, and let λ be the length measure on T such that $d_T(v_1, v_2) = \lambda(P_{v_1, v_2})$. We define T_v as the set of nodes contained in the subtree of T rooted at $v_r \in V$, i.e., $T_v = \{v' \in V : v \in P_{v_r, v'}\}$. Given the metric space (T, d_T) and measures μ and $\rho \in \mathcal{P}(T)$, Theorem 2.1 provides a closed-form expression for the 1-Wasserstein distance $\mathrm{W}_T(\mu, \rho)$.

Theorem 2.1. *Given two measures μ, ρ supported on $T = (V, E)$ with metric d_T, we have*

$$\mathrm{W}_T(\mu, \rho) = \sum_{e \in E} w_e |\mu(T_{v_e}) - \rho(T_{v_e})|, \quad (2)$$

where w_e is the weight of edge $e \in E$, and v_e is the node of $e \in E$ that is farther from the root (Le et al. 2019b).

From Theorem 2.1, we can compute the tree Wasserstein distance using a simple greedy matching algorithm where *mass* from measures μ and ρ is pushed from child to parent nodes and matched at parent nodes. This involves computing $|\mu(T_{v_e}) - \rho(T_{v_e})|$ for all nodes $\{v_e : e \in E\}$. We denote the optimal coupling associated with the tree Wasserstein distance as $\Pi^T_{\mu,\rho} \in \Gamma(\mu, \rho)$. Theorem 2.1 also provides a natural embedding for probability distributions on a tree to the ℓ_1 space, as stated in Corollary 2.2. See Appendix A for discussion on tree Wasserstein.

Corollary 2.2. *Let $\mathcal{W}_T$ be the set of probability measures defined on a tree T. Then, the tree Wasserstein space can be isometrically embedded in the ℓ_1 space.*

3 Optimal Transport Regression in Ultrametric Spaces

Our goal is to learn a tree metric on a discrete point set such that its optimal transport distance approximates the measured optimal transport distances. We define an optimization problem on ultrametrics as a proxy for tree metrics.

Definition 3.1. *Consider the set $\mathcal{X}$. An ultrametric function $d_u : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ is a metric on $\mathcal{X}$ that also satisfies the strong triangle inequality, i.e.,*

$$\forall x, y, z \in \mathcal{X} : d_u(x, y) \leq d_u(x, z) \wedge d_u(y, z).$$

Any compact ultrametric space can be represented by a rooted tree denoted as $T = (v_r, V, E)$, where v_r is the root node and $\mathcal{X}$ is the set of leaves. This representation also includes a height function $h : V \to \mathbb{R}$ with the following property: if v is the lowest common ancestor of the leaves x and $y \in \mathcal{X}$ (denoted $\mathrm{LCA}(x, y)$), then $h(v) = d_u(x, y)$. Furthermore, we induce a weight on the edges of the rooted tree T as follows: given an edge $(v_1, v_2) \in E$ where v_1 is closer to the root (or v_1 is the parent of v_2), we let $w(v_1, v_2) = h(v_1) - h(v_2)$. Then, the weighted tree distance between x and y is related to the heights as follows:

$$\begin{aligned} d_T(x, y) &= 2 \cdot h(\mathrm{LCA}(x, y)) - h(x) - h(y) \\ &= 2 \cdot d_u(x, y) - d_u(x, x) - d_u(y, y), \end{aligned} \quad (3)$$

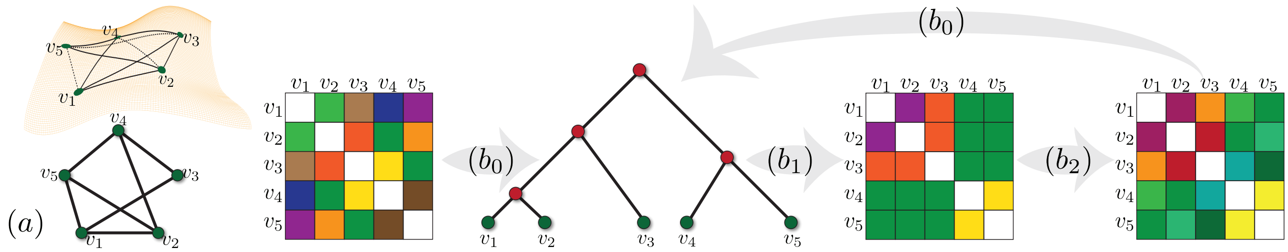

Figure 1: An overall summary of the our projected gradient descent procedure. (a) Measurements are points in a metric space, vertices of a weighted graph, or a distance matrix. (b_0) Hierarchical minimum spanning tree builds an ultrametric tree given a semimetric matrix. (b_1) We compute a distance matrix for the tree leaves. (b_2) We update the distance matrix by applying gradient descent on the optimal transport regression cost.

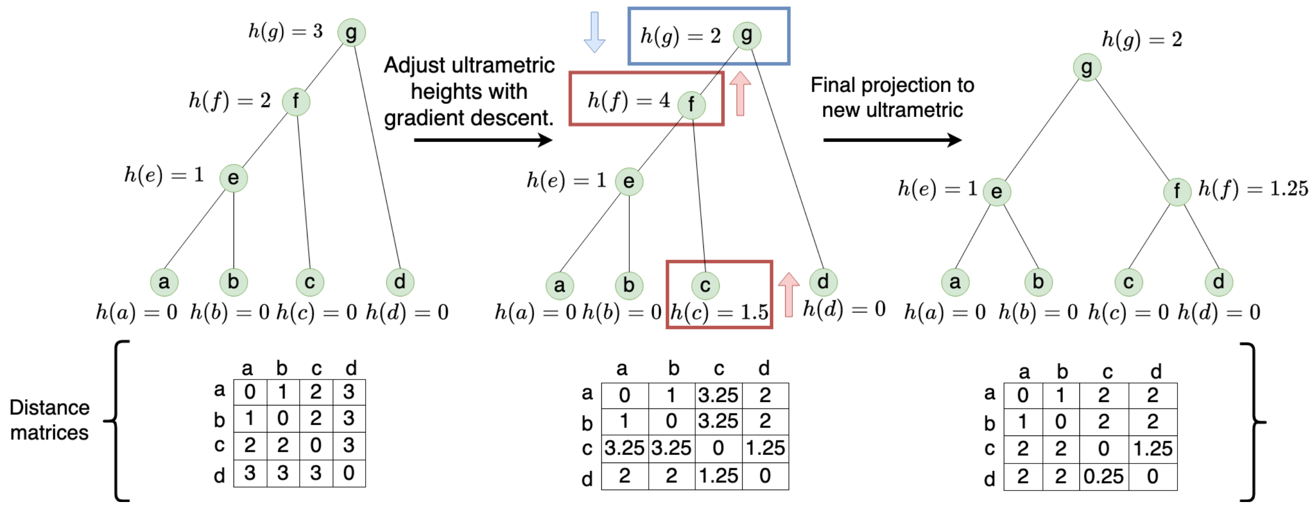

Figure 2: A stylized example of how the tree structure vary with heights of least common ancestors. Notice that in the second step, the heights of nodes f and c have increased while the height of node g has decreased. Due to these changes in height, the recomputed distance between nodes c and d is smaller than the recomputed distance between a and b so c and d are the first to be clustered in the final projection to the new ultrametric (instead of node c clustering with a and b in the original ultrametric).

i.e., $d_u(x, y) = \frac{1}{2} d_T(x, y)$. For a discrete ultrametric space $(\mathcal{X}, d_u)$, we compute the optimal transport distance as:

$$\mathrm{W}_u(\mu, \rho) = \langle \Pi^T_{\mu,\rho}, D_u \rangle$$

where $\Pi^T_{\mu,\rho} \in \Gamma(\mu, \rho)$ is the optimal coupling for the tree T representing the ultrametric space — constructed via the greedy matching described in Section 2 — and $D_u = (d_u(x_i, x_j))_{i,j\in[N]} \in \mathbb{R}^{N\times N}_+$. We now formalize the problem of learning an ultrametric on $\mathcal{X}$ for 1-Wasserstein distance approximation:

Problem 3.2. *Given an arbitrary discrete set $\mathcal{X}$ endowed with a semimetric, we want to find an ultrametric function $d_u : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ such that for a given set of distributions $\mathcal{S} \subseteq \mathcal{P}(\mathcal{X})$, we minimize the following cost function:*

$$C(d_u) = \sum_{\mu,\rho\in\mathcal{S}} \Big(\mathrm{W}(\mu, \rho) - \mathrm{W}_u(\mu, \rho)\Big)^2.$$

Problem 3.2 formulation allows for building an ultrametric on semimetric spaces such as metric spaces and positively weighted undirected graphs. Due to the nonconvex ultrametric constraint, the Problem 3.2 is nonconvex — in contrast to the optimization problem in (Yamada et al. 2022). Since $\mathcal{X}$ is a discrete set of size N (i.e. a metric space with N points), we represent $d_u : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ by $D_u \in \mathbb{UDM}_N$ — the set of ultrametric distance matrices of size N. With slight abuse of notation, we parameterize the cost in Problem 3.2 with D_u, i.e., $\min_{D_u\in\mathbb{UDM}_N} C(D_u)$. We then minimize the cost using projected gradient descent as follows:

$$D_u^{(k+1)} = \mathrm{proj}\big(D_u^{(k)} - \alpha \nabla C(D_u^{(k)})\big), \tag{4}$$

where α is the learning rate, $D_u^{(k)}$ is the ultrametric matrix at iteration k, and $\mathrm{proj} : \mathbb{R}^{N\times N} \to \mathbb{UDM}_N$ projects symmetric matrices to the space of ultrametric matrices of the same size; see Figure 1 for a summary. Entries of $D_u \in \mathbb{UDM}_N$ correspond to the pairwise distances between the points in an ultrametric space. The update formula above is a simplified version of the actual procedure; in practice, we do not perform unconstrained optimization over all $N \times N$ matrices. This point will be elaborated in **Computing the Gradient**.

Method	BBCSport	Twitter	RNAseq	USCA312	USAir97	Belfast
Ult. Tree	0.023 ± 0.010	0.062 ± 0.034	$\mathbf{0.041 \pm 0.002}$	$\mathbf{0.282 \pm 0.138}$	$\mathbf{0.043 \pm 0.018}$	$\mathbf{0.368 \pm 0.064}$
Quadtree	0.475 ± 0.012	0.507 ± 0.019	0.929 ± 0.003	N/A	N/A	N/A
Flowtree	0.175 ± 0.024	0.068 ± 0.036	0.079 ± 0.002	N/A	N/A	N/A
qTWD	0.070 ± 0.020	0.045 ± 0.012	0.092 ± 0.001	N/A	N/A	N/A
cTWD	0.028 ± 0.003	0.040 ± 0.029	0.044 ± 0.002	0.602 ± 0.142	0.122 ± 0.008	1.089 ± 0.117
Sliced-qTWD	0.065 ± 0.010	0.041 ± 0.016	0.091 ± 0.001	N/A	N/A	N/A
Sliced-cTWD	$\mathbf{0.020 \pm 0.002}$	$\mathbf{0.040 \pm 0.010}$	0.043 ± 0.002	0.674 ± 0.133	0.124 ± 0.006	1.00 ± 0.133
Sinkhorn, $\lambda = 1.0$	0.679 ± 0.700	0.716 ± 0.075	0.981 ± 0.160	0.813 ± 0.170	2.301 ± 0.144	0.713 ± 0.033

Table 1: Mean relative error $\left(\frac{|\mathrm{W}_1(\mu,\rho) - \mathrm{W}_T(\mu,\rho)|}{\mathrm{W}_1(\mu,\rho)}\right)$ $\pm$ standard deviation for 1-Wasserstein distance on real-world datasets.

A Projection Operator to Ultrametric Spaces. Given a semimetric, we use the standard Prim's algorithm to construct an ultrametric tree with leaf nodes in $\mathcal{X}$. We iteratively pick two closest points, e.g., $x_i, x_j \in \mathcal{X}$, and merge them into a new point $x_{(i,j)}$ — representing an internal node of the tree. The height of this node is equal to the pairwise distance of its children, i.e., $d_s(x_i, x_j)$. We repeat this process until we build a binary ultrametric tree, $T^u_{\mathcal{X}}$, with leaf vertices in $\mathcal{X}$. This hierarchical minimum spanning tree procedure serves as our projection, proj, from any semimetric matrix $D \in \mathbb{R}^{N\times N}$ to the ultrametric space $\mathbb{UDM}_N$.

Theorem 3.3. *The ultrametric matrix $D^*_u \in \mathbb{UDM}_N$ which is closest to D under the ℓ_∞ norm is $\mathrm{proj}(D) + \frac{1}{2}\|D - \mathrm{proj}(D)\|_\infty$ (Chepoi and Fichet 2000).*

Since the 1-Wasserstein distance on the ultrametric is *shift-invariant*, we opt to simply use $\mathrm{proj}(D)$.

Computing the Gradient. The cost $C(D_u)$ is a quadratic function of D_u. At iteration k, we fix the tree structure $T^{(k)}$ and compute the gradient of $C(D_u)$ over $\mathbb{R}^{N\times N}$, as follows:

$$\nabla C(D_u^{(k)}) = \sum_{\mu,\rho\in\mathcal{S}} \Big(\langle \Pi^{T^{(k)}}_{\mu,\rho} D_u^{(k)}\rangle - \mathrm{W}(\mu,\rho)\Big)\Pi^{T^{(k)}}_{\mu,\rho}, \quad (5)$$

where $\mathcal{S} \subseteq \mathcal{P}(\mathcal{X})$. We then adjust the tree structure in the next step — after projecting the updated distance matrix onto ultrametric matrices. Additionally, notice that the gradient computed in (5) implies each element of D_u is an independent variable. However, this matrix can only have $2N-1$ free parameters because it corresponds to an ultrametric tree where leaf nodes with the same least common ancestor (LCA) have the same distance, i.e., the height of their LCA. Therefore, we first parameterize D_u with $2N-1$ free variables associated with the height of each node in the tree, that is, $(D_u)_{i,j} = \theta_k$ if $\theta_k = h(\mathrm{LCA}(x_i, x_j))$ where $k \in [2N-1]$. In other words, we update both $(D_u)_{i,j}$ and $(D_u)_{i',j'}$ in the same way if $\mathrm{LCA}(x_i, x_j) = \mathrm{LCA}(x_{i'}, x_{j'})$. This is analogous to adjusting height parameters, $\{\theta_k\}_{k\in[2N-1]}$, in the ultrametric tree associated with D_u. After a gradient descent step, $\widehat{D} = D - \nabla_\theta C(D_u)$, diagonal elements of $\widehat{D}$ may no longer be zero. This is because $\widehat{D}_{i,i}$ is associated with the height of leaf node v_i on the tree, and leaf heights may deviate from their typical value of zero after each update. To get a valid distance matrix, we

Algorithm 1: Ultrametric tree optimization procedure

Input: discrete set $\mathcal{X}$, the distance matrix D for $\mathcal{X}$, learning rate α, maximum number of iterations t_{max}, training samples $\mathcal{S}$, minimum spanning tree algorithm MST
Output: An ultrametric d_u and associated tree $T^u_{\mathcal{X}}$

```
D_u^(0) = proj(D)
T_X^(0) = MST(D) (simultaneous during proj)
Let t = 0.
while t < t_max do
    Compute C(d_u) for training samples S.
    Given i, j ∈ [N], ∀i', j' ∈ [N] such that LCA(i, j) = LCA(i', j'), D̂_{i',j'} ← (D_u^(k) − α∇C(D_u^(k)))_{i',j'}
    ∀i, j ∈ [N], D̂_{i,j} ← 1/2 · (2 · D̂_{i,j} − D̂_{i,i} − D̂_{j,j}).
    D_u^{k+1} = proj(D̂)
    T_X^(k+1) = MST(D̂) (simultaneous during proj)
end while
return D_u^{t_max}, T_X^{t_max}
```

use equation (3) and convert new height parameters to ultrametric distances, viz., $\widehat{D}_{i,j} \leftarrow \frac{1}{2}\cdot(2\cdot\widehat{D}_{i,j} - \widehat{D}_{i,i} - \widehat{D}_{j,j})$ for all $i, j \in [N]$. We then use the updated matrix $\widehat{D}$ as the input for the projection to ultrametric space, proj. The parameterization based on LCA heights and the accompanying post-gradient processing based on equation (3) are the main distinctions to the simplified update rule in equation (4).

After a gradient descent step, the height of a parent node may become less than its children and the height of leaf nodes may change. This affects the distance matrix $\widehat{D}$ (see equation 3) in a way the causes the in the topology of estimated trees during training. See Figure 2 for an example of how the tree structure changes with the height of nodes.

Runtime Analysis. Since we construct the ultrametric (and associated tree) for each input matrix D using the hierarchical minimum spanning tree procedure, the number of nodes in the ultrametric tree is bounded by $O(N)$. Therefore, computing the optimal flow and the Wasserstein distance over the tree takes at most $O(N)$ time — linear in the number of points. However, computation complexity of the projection algorithm is $O(N^2)$ — a computational bottleneck for the proposed method. Therefore, the overall complexity of our training is $O(k \cdot N^2)$ where k is the number of gradient descent iterations. This training time is slower than that of (Yamada et al. 2022). Nevertheless, for a fixed discrete metric space $\mathcal{X}$, we only need to train the tree once and then we approximate Wasserstein distances in $O(N)$ time, i.e., the same as the inference time for (Yamada et al. 2022).

Why Use Ultrametrics? We use ultrametrics in Problem 3.2 as a proxy for general tree metrics. Given some metric d, the closest ultrametric d_u provides a 3-approximation for the tree metric to d (Agarwala et al. 1998). This is an upper bound on the distortion from the optimal tree metric caused by relaxing the problem to ultrametrics. This fact indicates that optimizing over the space of ultrametrics (as we do in Problem 3.2) is not overly restrictive compared to optimizing over the space of all tree metrics. Furthermore, ultrametrics are widely used in hierarchical clustering applications — most notably in bioinformatics to model phylogenetic trees. There is also an ultrametric that incurs a $\log n$ distortion with respect to Euclidean distance, i.e. $\|x - y\|_2 \leq d_u(x, y) \leq c \cdot \log n \|x - y\|_2$ (Fakcharoenphol, Rao, and Talwar 2003). Therefore, the ultrametric Wasserstein distance has a $\log n$ distortion compared to the original Euclidean 1-Wasserstein distance i.e. $\mathrm{W}_1(\mu, \rho) \leq \mathrm{W}_u(\mu, \rho) \leq \log n \mathrm{W}_1(\mu, \rho)$. However, this *does not* guarantee that the our projected gradient descent algorithm will achieve this $\log n$ distortion after training.

4 Experimental Results

We use PyTorch to implement our tree optimization algorithm, denoted *Ult. Tree*.[1] We compare the performance of our method with Sinkhorn distance (Cuturi 2013), Flowtree (Backurs et al. 2019), Quadtree (Indyk and Thaper 2003), weight optimized cluster tree Wasserstein distance (cTWD), weight optimized Quadtree Wasserstein distance (qTWD), and their sliced variants (Yamada et al. 2022). All reported results for Sinkhorn distance are computed with the Python Optimal Transport(Flamary et al. 2021) library and with regularization parameter $\lambda = 1.0$. We do not compare the performance of our method to the sliced tree Wasserstein distance of (Le et al. 2019b) since sliced-cTWD and sliced-qTWD consistently improve upon the results of (Le et al. 2019b). We not only compare the approximation accuracy in our experiments but also devise experiments which illustrate the benefits of changing tree structure throughout optimization compared to only learning weights for a fixed tree structure— as is the case in cTWD and qTWD methods (Yamada et al. 2022). Refer to Appendix C for more details regarding experiments, including dataset generation.

1-Wasserstein Approximations

Real-world Datasets. We compare 1-Wasserstein distance approximations for the Twitter and BBCSport word datasets (Huang et al. 2016) where training data consists of word frequency distributions per document. We also use three graph datasets: (1) USCA312, (2) USAir97 (Rossi and Ahmed 2015) and (3) the Belfast public transit graph (Kujala et al. 2018). Finally, we include a high-dimensional RNAseq dataset (publically available from the Allen Institute) which consists of vectors in $\mathbb{R}^{2000}$ (Yao et al. 2021). Additional details regarding datasets are in Appedix C. We summarize the error in Table 1 and the average runtime for each method in Table 2. Our ultrametric optimization method is slower than cTWD and qTWD in practice (although it has the same theoretical time complexity) because the size of the ultrametric tree generated by MST is larger than cTWD and qTWD.

Method	**USCA312**	**USAir97**	**Belfast**
Ult. Tree	0.00006	0.00013	0.00944
cTWD	**0.00004**	**0.00002**	**0.00009**
Sinkhorn, λ=1.0	0.01450	0.01883	0.4327
OT	0.00692	0.03819	0.5458
Method	**BBCSport**	**Twitter**	**RNAseq**
Ult. Tree	0.4204	0.1310	0.0017
Quadtree	**0.0001**	**0.0001**	**0.0001**
Flowtree	0.0002	0.0001	0.0001
cTWD	0.0004	0.0002	0.0002
Sinkhorn, λ=1.0	2.0851	0.3601	7.612
OT	1.6001	0.3820	0.570

Table 2: Average runtime for computing 1-Wasserstein distance of each algorithm based on their CPU implementations. The time complexity of the Quadtree-based methods, e.g., qTWD and sliced-qTWD, are similar to the Quadtree.

We outperform all other methods for the three graph datasets. However, for the two word datasets, Twitter and BBCSport, our error is worse compared to cTWD, qTWD, and their sliced variants. In the word datasets, word frequency distributions generated from documents have limited supports. It seems that this sparsity hinders the proper optimization of our projected gradient descent method as the constrained support of pairs of sparse distributions restricts each step of our optimization to a small portion of the tree. This is not an issue for graph datasets, since we use randomly generated full support distributions for training. In all cases, we outperform the distribution-agnostic methods like Quadtree and Flowtree.

Quadtree approximations *require* embeddings in a Euclidean space. However, for graph datasets, there are no faithful embbeddings for the nodes in a Euclidean space which preserves the semimetric information (from the edge weights). Therefore, we do not conduct experiments with Quadtree-based methods on graph datasets. In contrast, our proposed method, (sliced-)cTWD, and the Sinkhorn algorithms do not rely on the existence of such embeddings.

Synthetic Datasets. To compare the performance of our ultrametric tree optimization with all aforementioned algorithms, we generate 100 random point sets from the uniform distribution in a d-dimensional hypercube, e.g., $[-10, 10]^d$, where $d \in \{2, 5, 8, 11, 17\}$, and select 200 full-support dis-

[1]All code is publicly available at github.com/chens5/tree-learning.

Method	dim=2	dim=5	dim=8	dim=11	dim=14	dim=17
Ult. Tree	**0.104 ± 0.074**	**0.029 ± 0.023**	**0.016 ± 0.013**	**0.015 ± 0.011**	**0.012 ± 0.009**	**0.011 ± 0.007**
Flowtree	0.522 ± 0.145	0.559 ± 0.067	0.472 ± 0.053	0.430 ± 0.039	0.378 ± 0.030	0.348 ± 0.029
Quadtree	5.336 ± 1.046	2.981 ± 0.275	1.852 ± 0.158	1.310 ± 0.093	0.823 ± 0.036	0.624 ± 0.036
qTWD	0.557 ± 0.163	0.371 ± 0.056	0.297 ± 0.037	0.267± 0.031	0.220 ± 0.022	0.219 ± 0.020
cTWD	0.563 ± 0.177	0.371 ± 0.056	0.297 ± 0.037	0.267 ± 0.032	0.221 ± 0.022	0.219 ± 0.020
Sliced-qTWD	0.567 ± 0.132	0.387 ± 0.043	0.288 ± 0.026	0.260 ± 0.022	0.218 ± 0.016	0.206 ± 0.014
Sliced-cTWD	0.564 ± 0.138	0.387 ± 0.043	0.288 ± 0.026	0.261 ± 0.022	0.219 ± 0.016	0.206 ± 0.015
Sinkhorn, $\lambda = 1.0$	0.587 ± 0.257	0.071 ± 0.053	0.063 ± 0.046	0.074 ± 0.062	0.068 ± 0.049	0.076 ± 0.057

Table 3: Mean relative error for approximating 1-Wasserstein distance on a synthetic dataset of 100 randomly sampled from the uniform distribution over $[-10, 10]^d$. The training data consists of 200 randomly generated distribution pairs.

tribution pairs. In Section 4, we summarize the mean relative error for approximating the 1-Wasserstein distances.

Across all dimensions, our learned ultrametric method yields better approximations compared to all other methods by orders of magnitude. Furthermore, as our learned ultrametric optimization procedure does not depend on the embedding of the points themselves and only depends on the distance matrix, the variance of error does not vary with the points' dimension. The approximation bounds for Flowtree and Quadtree depends on the dimension and varies (in average) as the dimension changes. Moreover, our ultrametric tree approximation method is faster than the Sinkhorn algorithm (with $\lambda = 1.0$) while yielding a better quality optimal transport approximation.

Changing Tree Structure

One of the main advantages of our method, compared to qTWD and cTWD, is its capability to alter tree topology during training. In what follows, we show the utility of this property in both approximating 1-Wasserstein distance and recovering hidden tree metrics. In these experiments, given a tree (metric space) $\mathcal{X}$, we want to approximate 1-Wasserstein distance between distributions on its leaves. We purposefully initialize both our ultrametric learning procedure and cTWD with an incorrect tree topology.

Let us begin with an illustrative example. Consider $\mathcal{X} = \{x_0, x_1, x_2, x_3\}$ with the following distance matrix

$$D = \begin{pmatrix} 0 & 4 & 4 & 2 \\ 4 & 0 & 2 & 4 \\ 4 & 2 & 0 & 4 \\ 2 & 4 & 4 & 0 \end{pmatrix} \in \mathbb{R}_+^{4\times 4}$$

This is the distance matrix for the metric space of a binary ultrametric tree T with first-level clusters $\{x_0, x_4\}$ and $\{x_1, x_2\}$. The goal is to approximate Wasserstein distances in the original space $\mathcal{X}$. We first generate 200 pairs of distributions anc compute their Wasserstein distances in $\mathcal{X}$ (with respect to D). We optimize the weights of cTWD with respect D and train our ultrametric tree algorithm.

Now suppose we initialize both algorithms, our ultrametric optimization and cTWD, with the following adversarial distance matrix D':

$$D' = \begin{pmatrix} 0 & 2 & 4 & 4 \\ 2 & 0 & 4 & 4 \\ 4 & 4 & 0 & 2 \\ 4 & 4 & 2 & 0 \end{pmatrix} \in \mathbb{R}_+^{4\times 4},$$

where D' is associated with the binary tree T' which first-level clusters $\{x_0, x_1\}$ and $\{x_2, x_3\}$. Given D' as the input distance matrix, $T' \neq T$ is the initial tree structure for both algorithms. In the case of cTWD, the tree remains *fixed* as T'. For cTWD, after optimizing the weights of T', the mean relative error between Wasserstein distance of pairs of distributions on the T (ground truth) and their approximations with cTWD is 0.193 ± 0.101. On the other hand, our ultrametric tree optimization algorithm changes the tree topology throughout the training, and the final output tree metric is the same as D. Therefore, the learned ultrametric tree fully recovers the target metric space — even given an adversarial start — and perfectly computes Wasserstein distances. Furthermore, the final estimate tree topology, from our algorithm, is indeed T.

We extend this example by taking 100 random matrices (elements are i.i.d. from uniform distribution on $[10, 20]$) as incorrect initializations for cTWD and our ultrametric tree optimization algorithms. For *all* 100 random trials, our ultrametric tree optimization algorithms correctly recovers the original tree weights and its topology. Therefore, the learned ultrametric tree Wasserstein distances are exactly the same as the measured distances on $\mathcal{X}$; thereby giving the zero error. In contrast, mean ± standard deviation of the error for the Wasserstein distance in $\mathcal{X}$ for cTWD is 0.178 ± 0.103. These examples demonstrate the significance of correctly estimating the tree topology on approximating the Wasserstein distances, as cTWD incurs error when initialized with incorrect tree structures. More importantly, this illustrates the efficacy of our learned ultrametric tree in recovering the precise tree topology, thus enabling accurate computation of Wasserstein distances.

Learning Topologies of Random Trees. In what follows, we design an experiment to show the impact of changing tree topology learning Wasserstein distances and highlight the importance of changing tree topology from *iteration to iteration* using our minimum spanning tree procedure.

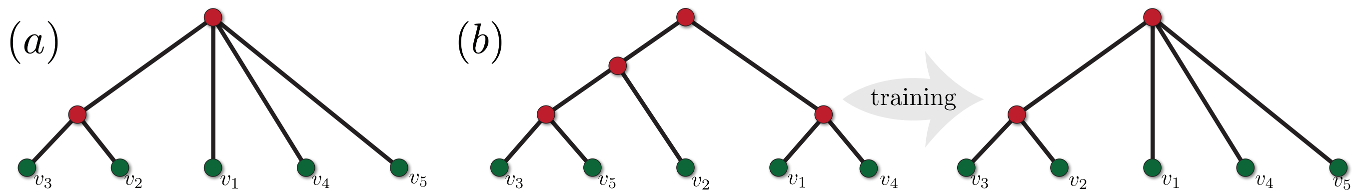

Figure 3: (a) : A randomly generated tree T with unit edge weights and leaf distance matrix D_T. (b) : We initialize the training process with noise-contaminated distance matrix, $\widetilde{D_T}$, and determine the initial tree topology on the left. We then train on true optimal transport distances and correctly recover the original tree structure on the right.

Method	σ	$\text{dist}(\widehat{T}, T)$	W_1 **error**
Ult. tree	5	$\mathbf{0.559 \pm 0.218}$	$\mathbf{0.262 \pm 0.102}$
skip-MST		1.043 ± 0.323	0.279 ± 0.102
cTWD		1.096 ± 0.281	0.285 ± 0.212
Ult. tree	4	$\mathbf{0.536 \pm 0.183}$	$\mathbf{0.275 \pm 0.257}$
skip-MST		0.992 ± 0.357	0.286 ± 0.109
cTWD		1.084 ± 0.315	0.295 ± 0.113
Ult. tree	3	$\mathbf{0.528 \pm 0.213}$	$\mathbf{0.265 \pm 0.092}$
skip-MST		0.983 ± 0.268	0.281 ± 0.098
cTWD		1.175 ± 0.388	0.270 ± 0.118
Ult. tree	2	$\mathbf{0.506 \pm 0.179}$	$\mathbf{0.268 \pm 0.095}$
skip-MST		0.815 ± 0.280	0.283 ± 0.108
cTWD		1.226 ± 0.358	0.293 ± 0.104

Table 4: Tree metric error and Wasserstein error for the learned ultrametric tree and the skip-MST method given an initial tree metric perturbed by Gaussian noise.

In this experiment, we adjust our ultrametric optimization method to only learn the weights of a fixed tree structure and change tree structure once at the last iteration, resulting in a more efficient procedure by avoiding the $O(N^2)$ complexity of projecting to the ultrametrics at each iteration. We initialize the tree topology using the hierarchical minimum spanning tree algorithm. Throughout training, we update the entries of the distance matrix via gradient descent without projecting onto the ultrametric space. At the end of training, we project the resulting distance matrix to the ultrametric space. This is what we call the *skip-MST method* as we avoid the minimum spanning tree procedure throughout training.

To compare cTWD, the skip-MST and the regular ultrametric optimization methods, we randomly generate 100 weighted tree metrics, from random trees with unit weights and 20 to 40 nodes. The distance matrix between leaf nodes is then perturbed by a symmetric noise matrix with i.i.d. elements from $\mathcal{N}(0, 2\sigma^2)$. We use this noise-contaminated distance matrix to determine the initial tree topology for all methods. We then synthetically generate probability distributions over the leaves of the underlying tree metrics and compute their exact optimal transport distances. We quantify the quality of the estimated tree with the following metric:

$$\text{dist}(\widehat{T}, T) = \frac{2}{N(N-1)} \|D_{\widehat{T}} - D_T\|_F,$$

where T is the original tree, D_T is the noiseless distance matrix, and $D_{\widehat{T}}$ is the distance matrix for estimated tree $\widehat{T}$. We report the results in Section 4.

For each value of $\sigma \in \{2, 3, 4, 5\}$, the difference between the final and true tree metrics for our ultrametric optimization algorithm is lower than that of both skip-MST and cTWD. This shows that ultrametric optimization adjusts the tree topology to correct for the inaccurate initial topology throughout training. Additionally, an incorrect tree topology has a detrimental effect on the estimated tree metric and optimal transport distance approximation as our ultrametric tree optimization algorithm shows between 4% and 6% improvement on Wasserstein error compared to the skip-MST and between 3% and 10% improvement compared to cTWD. Furthermore, it is noteworthy that, in most cases, skip-MST manages to achieve a lower error compared to cTWD even though, during training, it uses a fixed initial tree topology. In Figure 3, we illustrate an instance of a simple tree and how our ultrametric optimization procedure recovers the underlying tree topology from a bad initialization.

5 Conclusion

We present a novel contribution to the existing lines of work on linear-time optimal transport approximations. We propose a projected gradient descent algorithm that minimizes an optimal transport regression cost and relies on performing optimization in ultrametric space as a proxy for learning the optimal tree metric. The learned ultrametric tree offers an improved approximation for optimal transport distance compared to state-of-the-art tree Wasserstein approximations methods on synthetic datasets and provides a valuable way to recover tree topology. We acknowledge that the projection to ultrametric space is a computational bottleneck for our algorithm. An interesting future direction could be to find a fast projection to ultrametric space. Additionally, optimal transport on trees corresponds to an L_1 embedding so our learned tree can be used in locality sensitive hashing data structures for fast nearest neighbor queries. Finally, as our method adjusts tree topology, it is useful in applications where the data has some unknown underlying tree topology.

Acknowledgements

The authors would like to thank all anonymous reviewers for their valuable feedback. This work is partially supported by the NSF under grants CCF-2112665, CCF-2217033, and CCF-2310411.

References

Agarwala, R.; Bafna, V.; Farach, M.; Paterson, M.; and Thorup, M. 1998. On the approximability of numerical taxonomy (fitting distances by tree metrics). *SIAM Journal on Computing*, 28(3): 1073–1085.

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein generative adversarial networks. In *International conference on machine learning*, 214–223. PMLR.

Backurs, A.; Dong, Y.; Indyk, P.; Razenshteyn, I.; and Wagner, T. 2019. Scalable nearest neighbor search for optimal transport. *arXiv preprint arXiv:1910.04126*.

Chepoi, V.; and Fichet, B. 2000. l_∞-Approximation via subdominants. *Journal of mathematical psychology*, 44(4): 600–616.

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

Dominitz, A.; and Tannenbaum, A. 2009. Texture mapping via optimal mass transport. *IEEE transactions on visualization and computer graphics*, 16(3): 419–433.

El Moselhy, T. A.; and Marzouk, Y. M. 2012. Bayesian inference with optimal maps. *Journal of Computational Physics*, 231(23): 7815–7850.

Evans, S. N.; and Matsen, F. A. 2012. The phylogenetic Kantorovich–Rubinstein metric for environmental sequence samples. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(3): 569–592.

Fakcharoenphol, J.; Rao, S.; and Talwar, K. 2003. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, 448–455.

Flamary, R.; Courty, N.; Gramfort, A.; Alaya, M. Z.; Boisbunon, A.; Chambon, S.; Chapel, L.; Corenflos, A.; Fatras, K.; Fournier, N.; Gautheron, L.; Gayraud, N. T.; Janati, H.; Rakotomamonjy, A.; Redko, I.; Rolet, A.; Schutz, A.; Seguy, V.; Sutherland, D. J.; Tavenard, R.; Tong, A.; and Vayer, T. 2021. POT: Python Optimal Transport. *Journal of Machine Learning Research*, 22(78): 1–8.

Frogner, C.; Zhang, C.; Mobahi, H.; Araya, M.; and Poggio, T. A. 2015. Learning with a Wasserstein loss. *Advances in neural information processing systems*, 28.

Genevay, A.; Cuturi, M.; Peyré, G.; and Bach, F. 2016. Stochastic optimization for large-scale optimal transport. *Advances in neural information processing systems*, 29.

Huang, G.; Guo, C.; Kusner, M. J.; Sun, Y.; Sha, F.; and Weinberger, K. Q. 2016. Supervised word mover's distance. *Advances in neural information processing systems*, 29.

Indyk, P.; and Thaper, N. 2003. Fast image retrieval via embeddings. In *3rd international workshop on statistical and computational theories of vision*, volume 2, 5. Nice, France.

Kolouri, S.; Park, S. R.; Thorpe, M.; Slepcev, D.; and Rohde, G. K. 2017. Optimal mass transport: Signal processing and machine-learning applications. *IEEE signal processing magazine*, 34(4): 43–59.

Kujala, R.; Weckström, C.; Darst, R. K.; Mladenović, M. N.; and Saramäki, J. 2018. A collection of public transport network data sets for 25 cities. *Scientific data*, 5(1): 1–14.

Lavenant, H.; Claici, S.; Chien, E.; and Solomon, J. 2018. Dynamical optimal transport on discrete surfaces. *ACM Transactions on Graphics (TOG)*, 37(6): 1–16.

Le, T.; Huynh, V.; Ho, N.; Phung, D.; and Yamada, M. 2019a. Tree-wasserstein barycenter for large-scale multilevel clustering and scalable bayes. *arXiv preprint arXiv:1910.04483*.

Le, T.; Yamada, M.; Fukumizu, K.; and Cuturi, M. 2019b. Tree-sliced variants of Wasserstein distances. *Advances in neural information processing systems*, 32.

Lee, J.; and Raginsky, M. 2018. Minimax statistical learning with wasserstein distances. *Advances in Neural Information Processing Systems*, 31.

Montavon, G.; Müller, K.-R.; and Cuturi, M. 2016. Wasserstein training of restricted Boltzmann machines. *Advances in Neural Information Processing Systems*, 29.

Panaretos, V. M.; and Zemel, Y. 2016. Amplitude and phase variation of point processes. *The Annals of Statistics*, 44(2): 771–812.

Pele, O.; and Werman, M. 2009. Fast and robust earth mover's distances. In *2009 IEEE 12th international conference on computer vision*, 460–467. IEEE.

Rabin, J.; Peyré, G.; Delon, J.; and Bernot, M. 2011. Wasserstein barycenter and its application to texture mixing. In *International Conference on Scale Space and Variational Methods in Computer Vision*, 435–446. Springer.

Reich, S. 2013. A nonparametric ensemble transform method for Bayesian inference. *SIAM Journal on Scientific Computing*, 35(4): A2013–A2024.

Rossi, R. A.; and Ahmed, N. K. 2015. The Network Data Repository with Interactive Graph Analytics and Visualization. In *AAAI*.

Rubner, Y.; Tomasi, C.; and Guibas, L. J. 2000. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2): 99–121.

Solomon, J.; De Goes, F.; Peyré, G.; Cuturi, M.; Butscher, A.; Nguyen, A.; Du, T.; and Guibas, L. 2015. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Transactions on Graphics (ToG)*, 34(4): 1–11.

Solomon, J.; Rustamov, R.; Guibas, L.; and Butscher, A. 2014. Wasserstein propagation for semi-supervised learning. In *International Conference on Machine Learning*, 306–314. PMLR.

Takezawa, Y.; Sato, R.; Kozareva, Z.; Ravi, S.; and Yamada, M. 2021. Fixed support tree-sliced Wasserstein barycenter. *arXiv preprint arXiv:2109.03431*.

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

Yamada, M.; Takezawa, Y.; Sato, R.; Bao, H.; Kozareva, Z.; and Ravi, S. 2022. Approximating 1-Wasserstein Distance with Trees. *arXiv preprint arXiv:2206.12116*.

Yao, Z.; van Velthoven, C. T.; Nguyen, T. N.; Goldy, J.; Sedeno-Cortes, A. E.; Baftizadeh, F.; Bertagnolli, D.; Casper, T.; Chiang, M.; Crichton, K.; et al. 2021. A taxonomy of transcriptomic cell types across the isocortex and hippocampal formation. *Cell*, 184(12): 3222–3241.