PRIVATE SELECTION FROM PUBLIC CANDIDATES WITH WASSERSTEIN GUARANTEES

Anonymous authors
Paper under double-blind review

ABSTRACT

Given a sensitive dataset D and a public candidate set C of size k belonging to a metric space $(\mathcal{X}, d)$, we study how to select a size m multiset supported on C that approximately minimizes the 1-Wasserstein distance between its empirical distribution and that of D, under ε-differential privacy. Existing approaches typically rely on nearest-neighbor histograms over C, treating candidates essentially as unrelated bins and ignoring the geometry induced by d, despite its central role in 1-Wasserstein distance. At the same time, the number of possible size-m multisets $\binom{k+m-1}{m}$ grows combinatorially, making generic exact approaches computationally prohibitive. We therefore develop efficient geometry-aware algorithms. The main insight is to approximate the distance d on C by a randomized tree metric d_T with controlled expected distortion. This tree metric is induced by a weighted tree T whose leaves correspond to the candidates in C. The distance d_T between two candidates is their shortest-path distance in T, while the tree levels define increasingly fine partitions of C, from all candidates at the root to singletons at the leaves. Under d_T, 1-Wasserstein distance decomposes hierarchically across the tree, enabling two efficient ε-differentially private algorithms based on dynamic programming. The first exactly samples a size-m multiset from the exponential mechanism with score given by its negative empirical 1-Wasserstein distance to D under d_T, while the second privately estimates hierarchical subtree masses and post-process them into a compatible size-m multiset. The two mechanisms have similar worst-case utility guarantees, but our experiments characterize three distinct regimes where Tree-EM, hierarchical tree projection, and the nearest-neighbor baseline each outperform the others, demonstrating that our methods are complementary and present substantial gains whenever the target distribution has geometric structure that can be exploited.

1 INTRODUCTION

Real world datasets often contain sensitive information about individuals and therefore cannot be directly released or used for downstream tasks. Synthetic data offers a potential alternative: rather than using the sensitive dataset itself, one constructs an artificial dataset that preserves useful statistical properties while limiting the information revealed about any individual. Differential privacy (DP) provides a principled framework for achieving this goal and has become a standard notion of privacy in data analysis (Dwork et al., 2006). One common approach to generating differentially private synthetic data is to privately train a generative model, for example using DP stochastic gradient descent (Abadi et al., 2016), and then sample synthetic observations from it (Dockhorn et al., 2023; Xie et al., 2018). However, privately training deep generative models requires extensive hyperparameter tuning, and compute overhead due to per-example gradient clipping and noise addition. This has motivated work on training-free algorithms for DP synthetic data (Lin et al., 2023; Amin et al., 2024).

In this work, we study a problem motivated by this complementary training-free approach: using a public generative model to produce a large public dataset C of size k, and then privately selecting a multiset containing m of its records (with repetition) that best approximates the sensitive dataset D in terms of the 1-Wasserstein distance between their empirical distributions. Generation is entirely public and only the selection step depends on the sensitive dataset. This approach can be viewed as a one-shot version of Private Evolution (PE) (Lin et al., 2023), a recent training-free method for DP synthetic data that has gained significant attention (Lin et al., 2023; Xie et al., 2024; Lin et al., 2025; Hou et al., 2024; Swanberg et al., 2025; Zou et al., 2025; Hou et al., 2025). PE applies a similar idea

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

but iteratively: it generates public candidates, privately selects a fixed-size population that resembles the sensitive dataset, uses a public generative model to produce variations of the selected samples (for example, paraphrases of text records) and repeats. The problem we study could therefore serve as the private selection subroutine within PE.

Directly selecting a size-m multiset under the 1-Wasserstein distance provides two structural advantages over the standard nearest-neighbor (NN) histogram subroutine in PE: it handles larger candidate pools and explicitly exploits the geometry of the candidates. First, the NN subroutine privately estimates a probability distribution over the candidates using a noisy nearest-neighbor histogram and then samples m records from this distribution. Because it estimates a weight for every candidate, its worst-case privacy error can grow polynomially with the candidate-pool size k. In contrast, we directly select a size-m multiset, leading to only logarithmic dependence on k for fixed m. This makes our approach particularly well suited to the regime $k \gg m$, which naturally arises when selecting a small synthetic dataset from a large pool of public candidates.

The second advantage is that we formulate this selection problem using the 1-Wasserstein distance, which accounts for the geometry of the candidate space. In contrast, when estimating a nearest-neighbor histogram, candidates are only selected based on nearest-neighbor assignment counts, so candidates with the same number of votes are treated equally, regardless of their location in the data space.

The main computational difficulty is the fixed-size constraint. Directly minimizing the Wasserstein distance to D over all $\binom{k+m-1}{m}$ size-m multisets of C is a non-trivial combinatorial optimization problem, as naively searching the solution space takes exponential time. Our core contribution is a framework that makes this search tractable. We achieve this by approximating the metric on C with a randomized tree metric. This preserves Wasserstein distances up to a distortion factor, but makes certain computations efficient thanks to the tree structure. We leverage this idea to design two efficient, geometry aware randomized approximation algorithms for private selection. More concretely, our contributions are as follows:

- We reduce fixed-size Wasserstein selection over exponentially many multisets to polynomial-time computation. We achieve this by approximating the metric on the public candidates by a randomized tree metric and exploiting the resulting hierarchical decomposition of Wasserstein distance.
- By exploiting the hierarchical decomposition of the Wasserstein distance on randomized tree metrics, we develop two novel efficient algorithms: the first algorithm exactly samples from the exponential mechanism whose score is the negative Wasserstein distance under the tree metric. The second algorithm privately releases hierarchical nearest-neighbor counts over the tree and then post-processes them into a size-m multiset. We establish privacy, utility, runtime, and space guarantees for both methods. Crucially, the error due to privacy only scales polynomially with $\min\{m, k\}$, not with k as in naive DP histogram estimation.
- We provide experiments showing that nearest-neighbor histograms and both of our proposed algorithms are complementary: they can each be preferable on different instances.
- We also provide a support-adaptive bound which offers a potential explanation for why the private nearest-neighbor histogram can remain very strong when only a few candidates in C are nearest neighbors of records in D.

2 PRELIMINARIES

Notation. Let $(\mathcal{X}, d)$ be a metric data space, $D = (x_1, \ldots, x_n) \in \mathcal{X}^n$ be a sensitive dataset, and $C = \{c_1, \ldots, c_k\} \subset \mathcal{X}$ a public candidate set. Given a dataset A and a metric d, we denote $\mathrm{diam}_d(A) := \max_{a,a' \in A} d(a, a')$. We use the replacement adjacency notion where D and D' have the same size and differ by the replacement of a single record; we denote this as $D \sim D'$. A randomized mechanism M is ε-differentially private if, for every $D \sim D'$ and every measurable set S, $\mathbb{P}(M(D) \in S) \leq e^{\varepsilon}\mathbb{P}(M(D') \in S)$ (Dwork et al., 2006). For probability measures μ and ν on $\mathcal{X}$, define $W_{1,d}(\mu, \nu) := \inf_{\gamma \in \Pi(\mu,\nu)} \int d(x, y) d\gamma(x, y)$, where $\Pi(\mu, \nu)$ is the set of joint probability measures with marginals μ and ν. Let $\mu_D = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}$ be the empirical distribution of D; δ_x is the Dirac delta distribution.

Since the released records must belong to C, we first associate each private record with its nearest neighbor in the public set. For this, let $\pi_C(x)$ be the nearest neighbor of x on C; ties are broken with a fixed public rule. Then, the nearest-neighbor (NN) histogram of D on C is defined as $h_D = \frac{1}{n}\sum_{i=1}^{n} \delta_{\pi_C(x_i)}$. We will represent a size-m multiset supported on C by a multiplicity vector from the set $\mathcal{A}_m = \left\{a \in \mathbb{Z}_{\geq 0}^k : \sum_{j=1}^k a_j = m\right\}$. We denote $N_m = |\mathcal{A}_m| = \binom{k+m-1}{m}$. The empirical distribution of the dataset represented by $a \in \mathcal{A}_m$ is denoted by $q_a = \frac{1}{m}\sum_{j=1}^{k} a_j \delta_{c_j}$. We assume $k \geq 2$ and $m \geq 1$ throughout; when $k = 1$, the unique feasible output is m copies of the only candidate and no privacy mechanism is needed.

Problem statement. Our ultimate goal is to approximate the empirical distribution μ_D by the empirical distribution of a size-m multiset supported on C. Restricting the output support to C already incurs the unavoidable error $A(D, C) := \frac{1}{n}\sum_{i=1}^{n} d(x_i, \pi_C(x_i))$. Indeed, Proposition 5.1 in (González Lara et al., 2025) shows that the NN histogram h_D is optimal among all probability measures supported on C: $A(D, C) = W_{1,d}(\mu_D, h_D) = \inf_{\mathrm{supp}(\nu)\subseteq C} W_{1,d}(\mu_D, \nu)$. We therefore separate this support-restriction error from the additional error due to selection, and study how to privately solve

$$\mathrm{OPT}_{d,m}(D) := \min_{a\in\mathcal{A}_m} W_{1,d}(h_D, q_a). \tag{1}$$

For every $a \in \mathcal{A}_m$, the triangle inequality gives $W_{1,d}(\mu_D, q_a) \leq A(D, C) + W_{1,d}(h_D, q_a)$, so approximately solving Equation (1) translates directly to controlling the final approximation error to the original dataset.

Tree Metric. The Fakcharoenphol–Rao–Talwar (FRT) construction (Fakcharoenphol et al., 2003) yields a (randomized) rooted weighted tree T, whose leaves correspond bijectively to the candidates in C. This tree induces a metric $d_T(c, c')$ on C given by the shortest path distance between c and c' on T. d_T satisfies $d(c, c') \leq d_T(c, c')$ and $\mathbb{E}_T[d_T(c, c')] \leq O(\log k)d(c, c')$ for all $c, c' \in C$, where d is the original metric of the data space.

The construction of T starts by assigning C to the root and then recursively constructs a sequence of refined partitions $\mathcal{P}_0, \ldots, \mathcal{P}_L$, progressing from the trivial partition $\mathcal{P}_0 = \{C\}$ to the singleton partition $\mathcal{P}_L$. The partitions are hierarchical: each node at level ℓ corresponds to a cell in $\mathcal{P}_\ell$, and its child nodes correspond to the sub-cells formed when that cell is refined in $\mathcal{P}_{\ell+1}$. For a node v, let $\Gamma(v) \subseteq C$ denote its associated cell (the set of leaves in the subtree rooted at v). Thus, $\Gamma(r) = C$ for the root r, and each leaf corresponds to a single candidate. Let w_v be the edge weight between a non-root node v and its parent, with $w_r = 0$. Each edge entering level $\ell \geq 1$ has the common weight w_ℓ, so $w_v = w_\ell$ for every node v at that level. Edges from the root to the first level have weight $w_1 = \mathrm{diam}_d(C) := \Delta$, and $w_{\ell+1} := w_\ell/2$ for $\ell \geq 1$. We denote $K_\ell := |\mathcal{P}_\ell|$ and define $\mathcal{L} := \{\ell \in \{1, \ldots, L\} : K_\ell \geq 2\}$ as the set of non-trivial levels. See Section A for complete construction details.We use a binary representation of the tree only for implementation. An FRT need not be binary, but Lemma 5 shows that it can be binarized while preserving the properties we need.

1-Wasserstein Distance on Trees. For any probability measure μ on C, let $\mu(v) := \sum_{c_j\in\Gamma(v)} \mu(c_j)$ denote the measure's total mass inside the subtree at v. Similarly, for a multiplicity vector $a \in \mathcal{A}_m$, let $a(v) := \sum_{c_j\in\Gamma(v)} a_j$, giving $q_a(v) = a(v)/m$. Because an optimal transport plan between μ and ν must push $|\mu(v) - \nu(v)|$ mass across the edge above v, the tree metric admits the exact closed form (Le et al., 2019):

$$W_{1,d_T}(\mu, \nu) = \sum_{v\neq r} w_v |\mu(v) - \nu(v)|. \tag{2}$$

This form allows efficient tree-algorithms to be used for Wasserstein distance computation. Levels with $K_\ell = 1$ contribute zero to this sum since their unique cell is C, which carries unit mass under both measures. By the FRT distortion guarantees, any fixed pair of probability measures μ, ν on C satisfies $W_{1,d}(\mu, \nu) \leq W_{1,d_T}(\mu, \nu)$ and $\mathbb{E}_T[W_{1,d_T}(\mu, \nu)] \leq O(\log k)W_{1,d}(\mu, \nu)$. Hence, instead of solving Equation (1) directly, we solve $\min_{a\in\mathcal{A}_m} W_{1,d_T}(h_D, q_a)$, and the solution approximates $\mathrm{OPT}_{d,m}(D)$ up to the metric distortion factor.

Related work. Training-free algorithms for DP synthetic data have recently been proposed in practice (Lin et al., 2023; Xie et al., 2024; Lin et al., 2025; Swanberg et al., 2025; Amin et al., 2024).

On the theoretical side, early work studies DP synthetic data on discrete domains, with accuracy guarantees in terms of the maximum discrepancy over a class of statistical queries (Hardt et al., 2012; Blum et al., 2013; Dwork et al., 2015; González et al., 2026). DP synthetic data with 1-Wasserstein accuracy guarantees has been studied in He et al. (2023); Boedihardjo et al. (2024); González Lara et al. (2025). In particular, He et al. (2023) propose an algorithm that privately estimates the number of sensitive data points contained in cells of hierarchical partitions of the data space and constructs synthetic data from these estimates. Similar hierarchical approaches have been used in practice for DP clustering, a closely related problem (Balcan et al., 2017; Cohen-Addad et al., 2022). We remark that these works partition the entire data space, whereas we construct a hierarchy only over the finite set of public candidates. Feldman et al. (2024) gives an algorithm for DP Wasserstein density estimation on finite metric spaces which is particularly closely related to our work. They also approximate the underlying metric using a tree metric. The main distinction is that we directly select a finite synthetic dataset of a prescribed size, whereas they estimate a DP distribution. Post-processing their estimated distribution to obtain a DP synthetic dataset does not directly yield our m-sample selection guarantees.

Efficient sampling from the exponential mechanism over a large output space is sometimes possible when the score function has structure that can be exploited. This approach has been used for DP quantiles (Gillenwater et al., 2021), DP top-k selection (Gillenwater et al., 2022), and fixed-parameter tractable algorithms for DP synthetic data (Ghazi et al., 2026).

3 Two geometry-aware tree mechanisms

In this section we use the tree representations of the NN histogram to develop tree-EM and a hierarchical histogram, two algorithms that exploit candidate geometry rather than treating the public candidates as unrelated bins, and can be implemented efficiently by dynamic programming.
Both procedures are pure ε-DP.

3.1 Tree exponential mechanism

Given $\varepsilon > 0$, the exponential mechanism (McSherry & Talwar, 2007) samples an element $\hat{e}$ from a finite set $J = \{e_1, \dots, e_N\}$ according to $\mathbb{P}(\hat{e} = e_i) \propto \exp\left(-\frac{\varepsilon f(D,e_i)}{2\Delta_f}\right)$, where $f(D,\cdot)$ is a score function depending on the sensitive dataset D and $\Delta_f := \max_{e\in J} \max_{D\sim D'} |f(D,e) - f(D',e)|$ is its sensitivity. This mechanism is ε-DP and returns $\hat{e}$ approximately minimizing the function: $\mathbb{E}[f(D,\hat{e})] \le \min_{e\in J} f(D,e) + \frac{2\Delta_f \log N}{\varepsilon}$ (see, e.g., (Dwork et al., 2014)).

Using the exponential mechanism for problem (1) gives $\mathbb{P}(\widehat{a} = a) \propto \exp\left(-\frac{\varepsilon W_{1,d}(h_D,q_a)}{2\,\mathrm{diam}_d(C)/n}\right)$. Indeed, $f(D,a) = W_{1,d}(h_D,q_a)$ has sensitivity at most $\mathrm{diam}_d(C)/n$. Hence, $\widehat{a}$ is ε-DP and $\mathbb{E}[W_{1,d}(h_D,q_{\widehat{a}})] \le \mathrm{OPT}_{d,m}(D) + \frac{2\,\mathrm{diam}_d(C)\log N_m}{n\varepsilon}$, where $N_m = \binom{k+m-1}{m}$ is the number of possible size-m multisets. The difficulty is computational: naively sampling from this distribution requires enumerating these N_m possible outputs.

To make sampling tractable, we replace d by the tree metric d_T introduced in Section 2 and apply the exponential mechanism with $f(D,\cdot) = W_{1,d_T}(h_D,\cdot)$. We refer to the resulting mechanism, given explicitly in Algorithm 1, as ***tree-EM***. Since d_T approximates d, its utility guarantee transfers back to the original metric up to the distortion of the tree embedding, as formalized in Theorem 1.

The high-level idea is to sample an allocation by first assigning m records to the root and then recursively splitting the records assigned to each internal node between its two children, until m records in total are assigned across the leaves. The assignments of these m records to the leaves encode the final multiset that we will output. The tree decomposition of W_{1,d_T} makes this possible through two passes: a bottom-up dynamic program computes the probabilities required for each split, after which a top-down pass samples the splits recursively.

Sampling. Rewriting the unnormalized probability of sampling $a \in \mathcal{A}_m$ with the exponential mechanism in terms of the expression presented in Equation (2) gives

$$\exp\left[-\frac{\varepsilon n}{2D_T} W_{1,d_T}(h_D,q_a)\right] = \prod_{v\neq r} \exp\left[-\frac{\varepsilon n}{2D_T} w_v \left| h_D(v) - \frac{a(v)}{m}\right|\right],$$

where $D_T = \mathrm{diam}_{d_T}(C)$. Thus, the unnormalized probability factors into local terms depending only on $a(v)$, the number of output records placed in the leaves in $\Gamma(v)$. This decomposition allows us to sample an allocation as follows. Start from the root r with $a(r) = m$. Then, recursively, if an internal node v with children u, z is assigned t records, assign s records to u and $t - s$ to z, where

$$\mathbb{P}(a(u) = s \mid a(v) = t) = \frac{Z_u(s) Z_z(t-s)}{\sum_{j=0}^{t} Z_u(j) Z_z(t-j)}.$$

Here, $Z_u(s)$ denotes the sum of the local weights over all allocations in the subtree rooted at u that place exactly s output records among the leaves in $\Gamma(u)$. Conditioning on $a(v) = t$, the choice $a(u) = s$ forces $a(z) = t - s$, which gives the sampling probability above. It remains to show that these quantities, and hence the sampling probabilities, can be computed efficiently.

Computing the sampling probabilities. The quantities $Z_v(t)$ can be computed efficiently via dynamic programming. If v is a leaf, there is a unique allocation of t records in its subtree, so $Z_v(t) = \psi_v(t) := \exp\left[-\frac{\varepsilon n}{2D_T} w_v \left|h_D(v) - \frac{t}{m}\right|\right]$. If v is an internal node with children u and z, every allocation of t records below v induces a unique split $s \in \{0, \ldots, t\}$, with s records below u and $t - s$ below z. Therefore,

$$Z_v(t) = \psi_v(t) \sum_{s=0}^{t} Z_u(s) Z_z(t-s).$$

Computing these quantities bottom-up yields $Z_v(t)$ for every node v and $t \in \{0, \ldots, m\}$, enabling the top-down sampling described above.

Theorem 1 (Guarantees for tree-EM)**.** *Let D be a sensitive dataset, T a binarized FRT tree constructed from a public set C and d_T the metric it induces. Recall $\mathcal{A}_m$ is the set of size-m multiplicity vectors and $N_m = |\mathcal{A}_m| = \binom{k+m-1}{m}$. The tree exponential mechanism detailed in Algorithm 1 runs in time $O(kn + k^2 + km^2)$, is ε-DP and its output a satisfies*

$$\mathbb{E}[W_{1,d_T}(h_D, q_a) \mid T] \le \min_{\hat{a} \in \mathcal{A}_m} W_{1,d_T}(h_D, q_{\hat{a}}) + \frac{2D_T}{n\varepsilon} \log N_m.$$

If T is sampled independently according to the FRT construction and we let $r = \min\{k, m\}$, then the complete mechanism remains ε-DP and

$$\mathbb{E}_{T,a}[W_{1,d}(\mu_D, q_a)] \le A(D, C) + O(\log k)\, \mathrm{OPT}_{d,m}(D) + O\left(\frac{r\, \mathrm{diam}_d(C) \log((k+m)/r)}{n\varepsilon}\right). \tag{3}$$

The terms in Equation (3) have distinct interpretations: $A(D, C)$ measures how well the public candidates cover the private data; $\mathrm{OPT}_{d,m}(D)$ is the nonprivate fixed-size approximation error of h_D; the $O(\log k)$ factor accounts for the tree distortion; and the final term is the cost of privacy. In particular, the privacy error depends on $\log(N_m)$, which is at most $O(r \log((k+m)/r)$, rather than depending on accurate estimation of all k histogram entries. The space and time complexity are discussed in Section B.3.

3.2 Hierarchical histogram with tree projection

We now introduce a complementary algorithm that has comparable theoretical guarantees to tree-EM and can improve over it in practice when coarse-scale information about the distribution of votes across candidates can be estimated accurately. For example, if the candidates form four clusters and three receive no votes, estimating the total mass of each cluster can reveal this structure more accurately than estimating the mass of every candidate individually.

Recall that the NN histogram h_D is supported on the candidates C, which are the leaves of the FRT tree T introduced in Section 2. h_D naturally induces histograms at every level of the tree. Since the nodes at level ℓ form a partition of C, the vector $(h_D(v))_{v \in \mathcal{P}_\ell}$ is a histogram over the level-ℓ clusters. Moving upward in the tree therefore aggregates the leaf masses into increasingly coarse groups of nearby candidates.

This multiscale representation can preserve signal that is difficult to recover from a noisy histogram over individual candidates. For example, the mass of h_D may be spread across many nearby leaves, so that each individual candidate receives only a few votes and is difficult to estimate privately, while a cluster containing them may aggregate their mass and therefore be estimated more accurately.

Our mechanism privately estimates the histograms at each level by adding independent Laplace noise at every level and then chooses a size-m multiset in C that matches these estimates jointly in terms of certain objective. We refer to this algorithm as hierarchical histogram with tree projection, and we explain it below. The full procedure can be found in Algorithm 2.

Hierarchical histogram estimation. First, we allocate a privacy budget $\varepsilon_\ell > 0$ to each tree level $\ell \in \mathcal{L}$, with $\sum_{\ell \in \mathcal{L}} \varepsilon_\ell = \varepsilon$. For every node $v \in \mathcal{P}_\ell$, estimate

$$y_v = h_D(v) + Z_v, \qquad Z_v \sim \mathrm{Lap}\left(\frac{2}{n\varepsilon_\ell}\right),$$

independently. Privacy follows from the Laplace mechanism.

Projection objective. The noisy masses need not be nonnegative or consistent across levels. We recover a size-m multiset by selecting the one that is closest to the noisy counts in terms of the following objective

$$\widehat{F}(a) := \sum_{\ell \in \mathcal{L}} w_\ell \sum_{v \in \mathcal{P}_\ell} \left| y_v - \frac{a(v)}{m} \right|, \qquad a \in \mathcal{A}_m, \tag{4}$$

where the weights w_ℓ are exactly those appearing in the tree-Wasserstein decomposition in Equation (2). This objective extends the tree-Wasserstein decomposition to noisy node masses that may be negative or hierarchically inconsistent. In particular, if y_v were replaced by the true mass $h_D(v)$, then Equation (4) would equal $W_{1,d_T}(h_D, q_a)$ exactly. We return a minimizer $\widehat{a} \in \arg\min_{a \in \mathcal{A}_m} \widehat{F}(a)$, projecting the noisy counts back into a valid assignment in $\mathcal{A}_m$.

Projection implementation. We leverage again the tree structure and dynamic programming to make this projection efficient. Suppose that exactly t of the m output records are placed on leaves below a node v. Similar to Section 3.1, we identify the local contribution of that node to the objective. In this case that is

$$\ell_v(t) := \begin{cases} w_v \left| y_v - t/m \right|, & \text{if } v \in \mathcal{P}_\ell \text{ for some } \ell \in \mathcal{L}, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Then we can minimize $\hat{F}$ via dynamic programming as follows. For a node v and $t \in \{0, \ldots, m\}$, let $\widehat{F}_v(t)$ be the minimum contribution of the subtree rooted at v to the noisy objective when exactly t records are placed on $\Gamma(v)$. We compute $\widehat{F}_r(m) = \min_{a \in \mathcal{A}_m} \widehat{F}(a)$ and recover an assignment $\widehat{a}$ attaining this minimum by storing the minimizing splits. The values $\widehat{F}_v(t)$ can be computed from the leaves upward. At a leaf that receives t records, all of them must be assigned to that candidate, so $\widehat{F}_v(t) = \ell_v(t)$. At a node v with children u and z, we choose how many of the t records are allocated to each child subtree:

$$\widehat{F}_v(t) = \ell_v(t) + \min_{s \in \{0,\ldots,t\}} \left\{ \widehat{F}_u(s) + \widehat{F}_z(t-s) \right\}. \tag{6}$$

At the root we require all m records to be allocated, so $\widehat{F}_r(m)$ is the minimum of $\widehat{F}$ over all size-m multisets. Computing the tables bottom-up therefore gives the optimal objective value and the assignment achieving it.

Finally, we briefly discuss two aspects of the accuracy guarantee. First, the key way in which we exploit the fixed size constraint on the output space is by noting that the supports of $\hat{a} \in \arg\min_{a \in \mathcal{A}_m} \hat{F}(a)$ and $a^\star \in \arg\min W_{1,d_T}(h_D, q_a)$ can differ in at most $\min\{k, 2m\}$ entries, since both encode size-m multisets. Hence, it is possible to prove that the error incurred by $\hat{a}$ does not scale with all the Laplace noises added to the counts, but only with the ones on the union of its support with the one of $a^\star$. This contrasts with private Wasserstein density estimation, where the goal is to estimate the full distribution Feldman et al. (2024). Second, the utility guarantee can be first

obtained for a generic privacy allocation across levels. Since deeper levels have smaller weights w_ℓ and hence affect less the Wasserstein error, we can afford having larger errors due to smaller privacy budgets. So it is possible to optimize the expression over all privacy budget splits across levels that satisfy $\sum_{\ell \in \mathcal{L}} \varepsilon_\ell = \varepsilon$ gives an optimal allocation which is used in our final algorithm. All quantities in this allocation are public.

Theorem 2 (Guarantees for hierarchical tree projection)**.** *Denote $r := \min\{k, m\}$ and fix a public raw FRT tree T with L levels. Then, Algorithm 2 is ε-DP and conditional on T its output $\widehat{a}$ satisfies*

$$\mathbb{E}[W_{1,d_T}(h_D, q_{\widehat{a}}) \mid T] \le \min_{a \in \mathcal{A}_m} W_{1,d_T}(h_D, q_a) + O\left(\frac{D_T\, r \log(ek/r)}{n\varepsilon}\right). \tag{7}$$

If T is sampled independently of D from the FRT distribution, the mechanism remains ε-DP and

$$\mathbb{E}_{T,\widehat{a}} W_{1,d}(\mu_D, q_{\widehat{a}}) \le A(D, C) + O(\log k)\, \mathrm{OPT}_{d,m}(D) + O\left(\frac{\mathrm{diam}_d(C)\, r \log(ek/r)}{n\varepsilon}\right). \tag{8}$$

The mechanism can be implemented in $O(nk + k^2 + kLm + km^2)$ time and $O(k^2 + kLm)$ space.

For $m \le k$, the tree-EM privacy term obeys $\log N_m = \log \binom{k+m-1}{m} = O(m \log(ek/m))$, so the two mechanisms have the same worst-case dependence on m, k, n, ε up to universal constants and the common tree-diameter scale. They achieve this rate through different mechanisms: tree-EM directly randomizes over feasible size-m outputs, whereas hierarchical tree projection privately estimates mass at multiple resolutions and then deterministically reconciles these estimates into a feasible size-m output. The latter can therefore exploit accurately estimated coarse-scale structure even when individual leaf masses are difficult to estimate. Neither guarantee implies pointwise dominance, consistent with the different empirical regimes observed in the next section.

4 Experiments

We compare tree-EM and hierarchical tree projection with private NN histogram baselines. All reported errors are evaluated in the original ground metric $W_{1,d}$, not in the tree metric. The first experiment illustrates the scaling of the error with a larger candidate-pool; the second keeps the public geometry fixed and changes only the NN histogram in order to isolate three qualitatively different regimes. Additional experimental details and baseline configurations are given in Section E.

Scaling with the candidate-pool size. We first study the regime $k \gg m$, where directly selecting a size-m multiset can have substantially better dependence on k than privately estimating the entire NN histogram. We set $n = 100$, $m = 30$, and $\varepsilon = 1$. The private points are drawn uniformly from a disk of radius 0.25, while the public candidates are drawn from a four-component Gaussian mixture. We vary the number of candidates from $k = 50$ to $k = 5000$.

As shown in Figure 1, all NN histogram variants deteriorate as k grows, while tree-EM and the hierarchical histogram with tree projection remain comparatively stable. This agrees with the distinction suggested by the theoretical guarantees: NN histogram methods privately estimate a distribution over all k candidates, whereas tree-EM directly selects a size-m multiset and has privacy error that depends only logarithmically on k for fixed m. The experiment therefore illustrates the potential benefit of using our algorithms when dealing with a large public candidate pool while releasing a comparatively smaller synthetic dataset.

Three complementary regimes. We next fix one public candidate set and vary only the NN histogram h_D. There are $k = 400$ candidates arranged in four well-separated clusters, with $n = 200$, $m = 20$, $\varepsilon = 1$, and twenty trials. The sensitive datasets induces three possible NN histograms: (i) one candidate per cluster receives 50 votes each while the rest of the candidates do not receive any; (ii) diffuse mass over 200 candidates, 50 candidates per cluster receive one nearest-neighbor vote; and (iii) diffuse mass over 50 candidates inside a single cluster, where in one cluster every candidate receives four votes while the rest of the clusters do not receive. The public geometry of the candidates is identical across the three experiments. Table 1 summarizes the performance of the three approaches on each experiment.

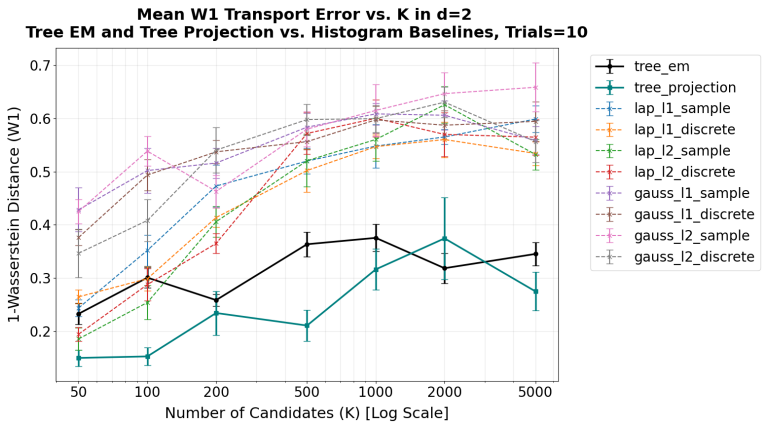


Figure 1: Wasserstein error as the public candidate pool grows. The private NN histogram baselines deteriorate with the increasing number of candidates, whereas both of our proposed algorithms remain more stable.

Input histogram	Best NN histogram	tree-EM	Tree projection
Four heavy bins	**0.640**	1.029	1.204
Diffuse across four clusters	1.754	**0.837**	1.343
Diffuse within one cluster	5.298	1.491	**0.853**

Table 1: Mean W_1 error over twenty trials. Only the NN histogram h_D changes across rows, while the candidates C remain fixed. Bold indicates the smallest mean error among the three mechanism families in that row.

We note that the DP NN histogram outperforms our algorithm when the votes are concentrated on a few candidates. We provide a potential explanation for this: the following lemma shows that the error of one of the DP NN histogram baselines implicitly adapts to the size of the support of h_D on the candidates, which can be $\min\{k, n\}$ in the worst case but could potentially be much smaller. The proof can be found in Appendix D.

Lemma 3 (Support adaptation of the NN histogram)**.** *Let $x = nh_D$ and let $\widetilde{h} = \Pi_2(x + Z)/n$, where Π_2 is Euclidean projection onto the count simplex and $Z = (Z_1, ..., Z_k) \overset{\text{iid}}{\sim} \text{Lap}(2/\varepsilon)$. Define $\tau_s(h_D)$ as the mass of h_D outside its s largest coordinates, let $R_m(p) = q_{\hat{a}(p)}$ for $\hat{a}(p) \in \arg\min_{a \in \mathcal{A}_m} \text{TV}(p, q_a)$, and let $t_m(h_D) := \min_{a \in \mathcal{A}_m} \text{TV}(h_D, q_a)$, where TV is the total variation distance. Then*

$$\mathbb{E}W_{1,d}(\mu_D, R_m(\widetilde{h})) \le A(D, C) + \text{diam}_d(C) \left[t_m(h_D) + 2 \inf_{1 \le s \le k} \left\{ \tau_s(h_D) + \frac{4sH_k}{n\varepsilon} \right\} \right],$$

where $H_k = \sum_{j=1}^{k} 1/j$. In particular, when h_D is s-sparse, the privacy contribution is $O(\text{diam}_d(C)s \log k/(n\varepsilon))$ rather than $O(\text{diam}_d(C)k/(n\varepsilon))$.

Overall, this experiment illustrates that the three approaches can exploit different forms of structure. One of the DP NN histogram baselines can benefit from sparsity of the support of h_D; tree-EM directly uses the geometry when selecting the complete size-m output; and hierarchical tree projection can exploit distributional structure that is visible at coarse levels of the candidate hierarchy even when the corresponding leaf-level histogram is diffuse. Consequently, the geometry-aware methods can substantially improve over NN histograms, but neither they nor the NN baseline uniformly dominate across instances.

5 Extensions and Discussion

The basic mechanisms use a single public tree and allow arbitrary candidate multiplicities. Multiple-tree and support-constraint extensions for tree-EM are given in Section F and briefly discussed below.

Multiple public trees. A single FRT draw has only an expected distortion guarantee, so an unlucky tree can distort the geometry more than typical. One can sample several public trees independently of the private data and treat the pair consisting of a tree index and a multiplicity vector as the output of one exponential mechanism. The root partition values provide the exact marginal weights needed to choose the tree, after which the usual top-down sampler generates the multiplicities. This avoids a separate private tree-selection step and therefore avoids privacy composition.

Constraints on the number of distinct selected candidates. Some applications may require the released size-m dataset to contain at least m' distinct candidates. The dynamic program can enforce this by augmenting its state with the number of distinct leaves used in each subtree. The resulting sampler is again exact for the exponential mechanism restricted to the constrained output space, with the same sensitivity and a privacy penalty involving the logarithm of the smaller constrained output space.

The hierarchical projection mechanism can also be run on any fixed public hierarchy but understanding how best to combine several public hierarchies is a difficult question because releasing counts on several trees would require additional privacy accounting.

Limitations and scope. Both tree mechanisms inherit the distortion of the public tree, and neither uniformly dominates the baselines. The hierarchical mechanism also spends privacy across several tree levels, so its practical behavior depends on how the useful geometry is distributed across scales. Our experiments are synthetic; evaluating these methods inside high-dimensional generation pipelines, including Private Evolution, remains an important next step.

6 Conclusion

We studied differentially private selection of exactly m records from a public candidate set under a 1-Wasserstein objective. The main difficulty is not only that the fixed-size output space is combinatorial, but also that standard NN histograms use the public candidates as unrelated bins even though Wasserstein loss depends strongly on their geometric structure. An FRT tree makes the problem computationally accessible. Tree-EM uses the tree decomposition to sample exactly from a geometry-aware exponential mechanism, while hierarchical tree projection privately estimates subtree masses and finds the size-m multiset most compatible with them. Both are polynomial-time, pure-DP mechanisms with similar worst-case utility guarantees. The experiments show that our methods can outperform the baselines when there is geometric structure to explore or the pool of candidates is too large. However, there are still some regimes where the baselines perform well, and we offer a plausible explanation for this.

Future work includes: (1) designing a rule to select the best method privately, since the regimes where different algorithms perform better depend on the private data, (2) designing an instance-adaptive algorithm and (3) evaluating our methods as a subroutine in real-world applications of private evolution.

AI use statement

In this work, we used generative AI tools to assist in the writing of proofs, design or provide feedback on research methodology or experiments and implement methods. We have not used generative AI tools for help develop theoretical models or conceptual frameworks, formulate mathematical claims, propose or refine hypotheses, or interpret results. Our work does not need AI to generate synthetic data sets, assist with translation, and clean and reformat datasets. We have reviewed AI-assisted work. For example, LLM-generated proofs were verified for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

Kareem Amin, Alex Bie, Weiwei Kong, Alexey Kurakin, Natalia Ponomareva, Umar Syed, Andreas Terzis, and Sergei Vassilvitskii. Private prediction for large-scale synthetic text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 7244–7262, 2024.

Maria-Florina Balcan, Travis Dick, Yingyu Liang, Wenlong Mou, and Hongyang Zhang. Differentially private clustering in high-dimensional euclidean spaces. In *International Conference on Machine Learning*, pp. 322–331. PMLR, 2017.

Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to noninteractive database privacy. *Journal of the ACM*, 60(2):1–25, 2013.

March Boedihardjo, Thomas Strohmer, and Roman Vershynin. Metric geometry of the privacy-utility tradeoff. *arXiv preprint arXiv:2405.00329*, 2024.

Vincent Cohen-Addad, Alessandro Epasto, Silvio Lattanzi, Vahab Mirrokni, Andres Munoz Medina, David Saulpic, Chris Schwiegelshohn, and Sergei Vassilvitskii. Scalable differentially private clustering via hierarchically separated trees. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 221–230, 2022.

Tim Dockhorn, Tianshi Cao, Arash Vahdat, and Karsten Kreis. Differentially Private Diffusion Models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=ZPpQk7FJXF.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pp. 265–284, 2006.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Cynthia Dwork, Aleksandar Nikolov, and Kunal Talwar. Efficient algorithms for privately releasing marginals via convex relaxations. *Discrete & Computational Geometry*, 53:650–673, 2015.

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

Vitaly Feldman, Audra McMillan, Satchit Sivakumar, and Kunal Talwar. Instance-optimal private density estimation in the wasserstein distance. *Advances in Neural Information Processing Systems*, 37:90061–90131, 2024.

Badih Ghazi, Cristóbal Guzmán, Pritish Kamath, Alexander Knop, Ravi Kumar, and Pasin Manurangsi. Fixed-parameter tractability of private synthetic data generation. *arXiv preprint arXiv:2606.11283*, 2026.

Jennifer Gillenwater, Matthew Joseph, and Alex Kulesza. Differentially private quantiles. In *International Conference on Machine Learning*, pp. 3713–3722. PMLR, 2021.

Jennifer Gillenwater, Matthew Joseph, Andres Munoz, and Monica Ribero Diaz. A joint exponential mechanism for differentially private top-k. In *International Conference on Machine Learning*, pp. 7570–7582. PMLR, 2022.

Tomás González, Cristóbal Guzmán, and Courtney Paquette. Mirror descent algorithms with nearly dimension-independent rates for differentially-private stochastic saddle-point problems. *SIAM Journal on Optimization*, 36(1):233–262, 2026.

Tomás González Lara, Giulia Fanti, and Aaditya Ramdas. Private evolution converges. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 148405–148441. Curran Associates, Inc., 2025. doi: 10.52202/085713-4961. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/da339ea9d0bcb0fc1a5ab07961f022bb-Paper-Conference.pdf.

Moritz Hardt, Katrina Ligett, and Frank Mcsherry. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

Yiyun He, Roman Vershynin, and Yizhe Zhu. Algorithmically effective differentially private synthetic data. In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 3941–3968. PMLR, 2023.

Charlie Hou, Akshat Shrivastava, Hongyuan Zhan, Rylan Conway, Trang Le, Adithya Sagar, Giulia Fanti, and Daniel Lazar. Pre-text: training language models on private federated data in the age of LLMs. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Charlie Hou, Mei-Yu Wang, Yige Zhu, Daniel Lazar, and Giulia Fanti. Private federated learning using preference-optimized synthetic data. *arXiv preprint arXiv:2504.16438*, 2025.

Tam Le, Makoto Yamada, Kenji Fukumizu, and Marco Cuturi. Tree-sliced variants of wasserstein distances. *Advances in neural information processing systems*, 32, 2019.

Zinan Lin, Sivakanth Gopi, Janardhan Kulkarni, Harsha Nori, and Sergey Yekhanin. Differentially Private Synthetic Data via Foundation Model APIs 1: Images. In *ICLR*, 2023.

Zinan Lin, Tadas Baltrusaitis, and Sergey Yekhanin. Differentially private synthetic data via APIs 3: Using simulators instead of foundation models. *arXiv preprint arXiv:2502.05505*, 2025.

Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pp. 94–103. IEEE, 2007.

Marika Swanberg, Ryan McKenna, Edo Roth, Albert Cheu, and Peter Kairouz. Is API access to LLMs useful for generating private synthetic tabular data? In *ICLR Workshop: Will Synthetic Data Finally Solve the Data Access Problem?*, 2025. URL https://openreview.net/forum?id=e4uwE1muhR.

Chulin Xie, Zinan Lin, Arturs Backurs, Sivakanth Gopi, Da Yu, Huseyin Inan, Harsha Nori, Haotian Jiang, Huishuai Zhang, Yin Tat Lee, Bo Li, and Sergey Yekhanin. Differentially private synthetic data via foundation model APIs 2: text. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

Tianyuan Zou, Yang Liu, Peng Li, Yufei Xiong, Jianqing Zhang, Jingjing Liu, Xiaozhou Ye, Ye Ouyang, and Ya-Qin Zhang. Contrastive Private Data Synthesis via Weighted Multi-PLM Fusion. *arXiv preprint arXiv:2502.00245*, 2025.

A FRT CONSTRUCTION AND BINARIZATION

This section gives the construction details behind the FRT reduction used in the main text and proves that the resulting tree can be binarized without changing its metric.

Definition 4 (Hierarchically separated tree metric)**.** A hierarchically separated tree (HST) over the candidate set C is a rooted weighted tree $T = (V, E)$ whose leaves are identified with the candidates in C. For each node $v \in V$, let $\Gamma(v) \subseteq C$ denote the subset of candidates in the subtree rooted at v:

$$\Gamma(v) = \{c \in C : c \text{ is a descendant leaf of } v\}.$$

In particular, $\Gamma(r) = C$ for the root r, and $\Gamma(v) = \{c\}$ if v is the leaf corresponding to $c \in C$.

For any measure μ supported on the leaves, we use the shorthand $\mu(v) := \sum_{c_j \in \Gamma(v)} \mu(c_j)$. For a multiplicity vector $a \in \mathcal{A}_m$, we likewise write $a(v) := \sum_{c_j \in \Gamma(v)} a_j$, so that $q_a(v) = a(v)/m$.

All edges between levels $\ell - 1$ and ℓ have the same weight, denoted by w_ℓ, and these weights decrease geometrically:

$$w_{\ell+1} = \frac{1}{2} w_\ell.$$

The induced tree metric on C is

$$d_T(c, c') = \sum_{e \in \mathrm{path}_T(c,c')} w_e,$$

where $\mathrm{path}_T(c, c')$ is the shortest path between the leaves corresponding to c and c', and w_e denotes the weight of edge e.

The goal of the FRT construction (Fakcharoenphol et al., 2003) is to build a tree whose levels induce progressively finer partitions $\mathcal{P}_0, \mathcal{P}_1, \ldots$ of the candidate set C, starting with the entire set at the root and recursively subdividing clusters until every candidate is an isolated leaf.

Let $\Delta := \mathrm{diam}_d(C)$ and choose the initial edge weight $w_1 = \Delta$, with $w_{\ell+1} := w_\ell/2$ for $\ell \geq 1$. The construction shares two global sources of randomness across all levels, sampled once at the beginning: a single random permutation π of the candidates C, and a random scale shift $\beta \sim \mathrm{Unif}[1, 2)$. Set the clustering radii to $r_1 := \beta w_1/4$ and $r_{\ell+1} := r_\ell/2$ for $\ell \geq 1$; equivalently, $r_\ell = \beta w_\ell/4$.

The root corresponds to the trivial partition $\mathcal{P}_0 = \{C\}$. At each subsequent level $\ell \geq 1$, the current partition $\mathcal{P}_{\ell-1}$ is refined by decomposing each cluster $S \in \mathcal{P}_{\ell-1}$ into smaller child clusters. Specifically, each candidate $c \in S$ is assigned to the first candidate $c' \in C$ in the order of π that falls within distance r_ℓ:

$$d(c, c') \leq r_\ell.$$

Candidates in S assigned to the same center c' form a child cluster of S, connected to S by an edge of length w_ℓ. The center c' may lie outside S; assignments are grouped separately within each parent cluster. Both the edge weights w_ℓ and the clustering radii r_ℓ halve at each level. This refinement continues until every cluster becomes a singleton, at which point each leaf is identified with a unique candidate in C.

The same random permutation and the same random shift $\beta \sim \mathrm{Unif}[1, 2)$ are used at every level (Fakcharoenphol et al., 2003). If T is the random tree obtained by this process and d_T denotes the induced tree metric, then $d_T(c, c') \geq d(c, c')$ for every pair of candidates, while

$$\mathbb{E}_T[d_T(c, c')] \leq O(\log k) d(c, c').$$

Every nonroot level-ℓ cluster has diameter at most $2r_\ell = \beta w_\ell/2 \leq w_\ell$. If two leaves first separate at level ℓ, their common parent has diameter at most $2w_\ell$ (also for $\ell = 1$, since $\Delta \leq w_1$), while their tree path contains two edges of length w_ℓ. This proves $d_T(c, c') \geq d(c, c')$. Since the edge weights decrease geometrically, $w_{\ell+1} = w_\ell/2$, every root-to-leaf path has length at most

$$\sum_{\ell \geq 1} w_\ell \leq 2w_1.$$

Consequently, every leaf-to-leaf distance is at most $4w_1$, and hence $\mathrm{diam}_{d_T}(C) = O(\mathrm{diam}_d(C))$.

The main text explains how these pairwise guarantees transfer to Wasserstein distance and states the additive tree-Wasserstein identity used by the dynamic program. We now justify the binarization step.

Lemma 5 (Binarization preserves the tree metric)**.** *Let T be a rooted weighted tree whose leaves are the candidates in C. There exists a rooted binary weighted tree T^{bin} with the same candidate leaves such that*

$$d_{T^{\mathrm{bin}}}(c, c') = d_T(c, c') \qquad \textit{for every } c, c' \in C.$$

Consequently, binarization does not change the diameter of the candidate set, the tree-Wasserstein distance between distributions supported on C, or the sensitivity of the tree-Wasserstein loss.

Proof. If T is binary, then we are done. If not, then we transform T in two steps.

First, we remove internal nodes having exactly one child. If the root has exactly one child, we delete the root and declare its child to be the new root; this changes no path between candidate leaves. Suppose that a non-root node v has parent p and exactly one child u. Let the edges (p, v) and (v, u) have lengths $w_{p,v}$ and $w_{v,u}$, respectively. We delete v and these two edges, and connect p directly to u with an edge of length $w_{p,u} = w_{p,v} + w_{v,u}$.

Any path that previously passed from p to u through v now uses the new edge (p, u), whose length is exactly the sum of the two deleted edge lengths. Therefore, every path between candidate leaves has the same length as before. We repeat this operation until every internal node has at least two children.

Second, we replace every node having more than two children by a binary structure. Suppose that a node v has children $u_1, \ldots, u_q$, where $q > 2$, and let w_i denote the length of the original edge from v to u_i. Connect v directly to u_1 with an edge of length w_1, and connect v to a new auxiliary node z_2 with an edge of length zero. Next, connect z_2 to u_2 with an edge of length w_2, and connect z_2 to another new auxiliary node z_3 with an edge of length zero. Continue in this way: for each

$i = 2, \dots, q-2$, connect z_i to u_i with an edge of length w_i and connect z_i to z_{i+1} with an edge of length zero. Finally, connect z_{q-1} to u_{q-1} and u_q with edges of lengths w_{q-1} and w_q, respectively.

Every node in the resulting structure has at most two children. Moreover, the path from v to each original child u_i consists of zero-length auxiliary edges followed by one edge of length w_i. Therefore, its total length is still w_i.

Now consider any two candidate leaves $c, c' \in C$. If both leaves belong to the same child subtree of v, the path between them is not affected by the replacement at v. Otherwise, suppose that c belongs to the subtree rooted at u_i and c' belongs to the subtree rooted at u_j, where $i \neq j$. Before binarization, the part of their path between u_i and u_j had length $w_i + w_j$. After binarization, this part contains the edge of length w_i leading to u_i, the edge of length w_j leading to u_j, and only zero-length auxiliary edges between them. Its length is therefore still $w_i + w_j$. Thus, the distance between c and c' is unchanged.

Applying this replacement to every node with more than two children produces a binary tree T^{bin} satisfying $d_{T^{\mathrm{bin}}}(c, c') = d_T(c, c')$, for all $c, c' \in C$.

Finally, the diameter and Wasserstein distance depend only on the distances between candidate leaves. Since all these distances are preserved, $\mathrm{diam}_{d_{T^{\mathrm{bin}}}}(C) = \mathrm{diam}_{d_T}(C)$ and $W_{1,d_{T^{\mathrm{bin}}}}(\mu, \nu) = W_{1,d_T}(\mu, \nu)$ for every pair of distributions μ, ν supported on C. In particular, the tree-Wasserstein loss and its sensitivity under changes to the private dataset are also unchanged. □

B Details on tree-EM

This section provides the missing details about Algorithm 1.

B.1 Algorithm

We present the exact algorithm described in the main body.

Algorithm 1 Tree exponential mechanism (tree-EM)

Require: Private dataset D, public binary tree T, output size m, and privacy parameter ε

Compute the NN histogram h_D and every subtree mass $h_D(v) = \sum_{c \in \Gamma(v)} h_D(c)$.
Set $w_r \leftarrow 0$, $D_T = \mathrm{diam}_{d_T}(C)$, and $\lambda = \varepsilon n / (2D_T)$.
Compute $\psi_v(t) = \exp[-\lambda w_v |h_D(v) - t/m|]$ for every node v and $t \in \{0, \dots, m\}$.
for each node v in bottom-up order **do**
    if v is a leaf **then**
        $Z_v(t) \leftarrow \psi_v(t)$ for every $t = 0, \dots, m$.
    else
        Let u, z be the children of v.
        $Z_v(t) \leftarrow \psi_v(t) \sum_{s=0}^{t} Z_u(s) Z_z(t-s)$ for every $t = 0, \dots, m$.
    end if
end for
Set $a(r) \leftarrow m$.
for each internal node v in top-down order **do**
    Let u, z be the children of v and set $t \leftarrow a(v)$.
    Sample $s \in \{0, \dots, t\}$ with probability $\dfrac{Z_u(s) Z_z(t-s)}{\sum_{j=0}^{t} Z_u(j) Z_z(t-j)}$.
    Set $a(u) \leftarrow s$ and $a(z) \leftarrow t - s$.
end for
return $a_j = a(c_j)$ copies of each candidate c_j.

B.2 Proof of Theorem 1

For compactness, write $L_T(D, a) := W_{1,d_T}(h_D, q_a)$ and $\lambda := \varepsilon n / (2D_T)$.

Proof. **Runtime.** The running-time claim follows from Lemma 6.

Exact sampling. For a node v and $t \in \{0, \dots, m\}$, let $\mathcal{A}_v(t)$ be the set of multiplicity assignments to the candidate leaves below v whose total count is t. For such an assignment, define the loss contributed by the subtree rooted at v as

$$L_v(a) = \sum_{y \text{ in the subtree rooted at } v} w_y \left| h_D(y) - \frac{a(y)}{m} \right|, \qquad w_r = 0.$$

We first prove, by induction from the leaves to the root, that

$$Z_v(t) = \sum_{a \in \mathcal{A}_v(t)} \exp[-\lambda L_v(a)]. \tag{9}$$

If v is a leaf, there is exactly one assignment with total count t, namely assigning t copies to that leaf. Therefore, $Z_v(t) = \psi_v(t) = \exp[-\lambda L_v(a)]$.

Now suppose that v is an internal node with children u and z. Every assignment in $\mathcal{A}_v(t)$ has a unique value $s = a(u) \in \{0, \dots, t\}$ and therefore assigns $t - s$ records below z. Conversely, an assignment below u with total s and an assignment below z with total $t - s$ uniquely determine an assignment below v. Its loss is

$$L_v(a) = w_v \left| h_D(v) - \frac{t}{m} \right| + L_u(a) + L_z(a).$$

Summing the exponential weight over all assignments and grouping them by s gives

$$\sum_{a \in \mathcal{A}_v(t)} e^{-\lambda L_v(a)} = \psi_v(t) \sum_{s=0}^{t} Z_u(s) Z_z(t-s) = Z_v(t),$$

proving (9).

We next prove that the top-down procedure samples from these weights. Given that node v receives total count t, the total weight of all assignments for which its first child receives count s is $\psi_v(t) Z_u(s) Z_z(t-s)$. Since $\psi_v(t)$ is the same for every possible split, the conditional probability of s is exactly

$$\frac{Z_u(s) Z_z(t-s)}{\sum_{j=0}^{t} Z_u(j) Z_z(t-j)},$$

which is the sampling rule in algorithm 1. Conditional on the split, the assignments inside the two child subtrees factor and are sampled recursively. Starting from $a(r) = m$ and using (9) at the root therefore gives

$$\Pr(\widehat{a} = a \mid T) = \frac{\exp[-\lambda L_T(D, a)]}{Z_r(m)}.$$

This proves the exact-sampling claim. Notice that the argument applies to every finite binary tree and does not require the tree to be complete, balanced, or of logarithmic depth.

Privacy. For privacy, if $D \sim D'$, then their NN histograms differ by moving mass $1/n$ between two leaves. Hence

$$W_{1,d_T}(h_D, h_{D'}) \le \frac{D_T}{n}.$$

By the reverse triangle inequality, for every $a \in \mathcal{A}_m$,

$$|L_T(D, a) - L_T(D', a)| \le W_{1,d_T}(h_D, h_{D'}) \le \frac{D_T}{n}.$$

Thus the loss sensitivity is at most D_T/n, and $\lambda = \varepsilon/(2(D_T/n))$ is precisely the exponential-mechanism calibration. The mechanism is therefore ε-differentially private.

Utility. The fixed-tree utility bound follows from the standard exponential-mechanism guarantee (McSherry & Talwar, 2007), applied to the output space $\mathcal{A}_m$ with cardinality N_m and sensitivity D_T/n.

It remains to prove the guarantee in the original metric. Given that the FRT tree is sampled independently of the private data, averaging the fixed-tree mechanism over the tree randomness preserves ε-differential privacy. The projection inequality gives

$$W_{1,d}(\mu_D, q_{\widehat{a}}) \le A(D, C) + W_{1,d}(h_D, q_{\widehat{a}}).$$

Since d_T dominates d, the second term is at most $L_T(D, \widehat{a})$. Applying the fixed-tree guarantee conditionally on T, and letting a^* minimize $W_{1,d}(h_D, q_a)$, gives

$$\mathbb{E}[W_{1,d}(\mu_D, q_{\widehat{a}})] \le A(D, C) + \mathbb{E}_T[L_T(D, a^*)] + \frac{2 \log N_m}{n\varepsilon} \mathbb{E}_T[D_T].$$

The FRT pairwise stretch guarantee, applied to an optimal coupling between h_D and q_{a^*}, yields

$$\mathbb{E}_T[L_T(D, a^*)] \le O(\log k)\, \mathrm{OPT}_{d,m}(D).$$

Moreover, by the diameter bound for the FRT construction, $D_T = O(\mathrm{diam}_d(C))$, and hence $\mathbb{E}_T[D_T] = O(\mathrm{diam}_d(C))$. Substituting these two bounds gives

$$\mathbb{E}_{T,\widehat{a}} W_{1,d}(\mu_D, q_{\widehat{a}}) \le A(D, C) + O(\log k)\, \mathrm{OPT}_{d,m}(D) + O(\mathrm{diam}_d(C)) \frac{\log N_m}{n\varepsilon}.$$

A simple bound on N_m finishes the proof. □

B.3 Runtime and space details

The main text states the cost of the dynamic program. Here we account for nearest-neighbor assignment, tree construction and preprocessing, and sampling separately.

Lemma 6 (Runtime and space complexity)**.** *Algorithm 1 can be implemented to run in time $O(nk + k^2 + km^2)$ and use $O(k^2 + km)$ space.*

We prove this lemma below by analyzing the runtime and space complexity of each step in the algorithm.

Nearest-neighbor assignment. Computing the nearest neighbor

$$\pi_C(x_i) \in \arg\min_{c \in C} d(x_i, c)$$

by exhaustive search requires $O(nk)$ distance evaluations. The resulting leaf histogram can be accumulated online using $O(k)$ memory, so it is unnecessary to store all nk pairwise distances. This computation is independent across the private records and is therefore straightforward to parallelize. In metric spaces admitting efficient exact or approximate nearest-neighbor data structures, the $O(nk)$ term may be reduced, but we retain exhaustive search as the general worst-case bound.

FRT tree construction. Let

$$\Phi = \frac{\max_{c,c' \in C} d(c, c')}{\min_{c \ne c'} d(c, c')}$$

be the aspect ratio of the finite candidate metric, and let

$$L = O(\log \Phi)$$

be the number of relevant dyadic scales in the FRT hierarchy. Computing and storing the dense candidate distance matrix requires $O(k^2)$ distance evaluations and $O(k^2)$ space. Given this matrix, a direct implementation of the FRT clustering procedure takes $O(k^2 L)$ time: at each of the L scales, each candidate may be compared with the candidates appearing before it in the random permutation. However, the extra factor L is not intrinsic: after the $O(k^2)$ pairwise distances are available, the FRT hierarchy can be constructed in $O(k^2)$ time by processing the relevant distance thresholds across scales rather than rescanning all pairs independently at every level (Fakcharoenphol et al., 2003). This is the implementation used in the runtime bound stated in Theorem 1.

The raw FRT hierarchy may contain unary chains corresponding to clusters that do not change between consecutive scales. These chains can be contracted by adding their edge lengths, without changing the induced distances between candidate leaves. After contracting unary nodes, the rooted

tree has k leaves and no internal node of degree one. It therefore contains $O(k)$ nodes. Any node of degree larger than two can then be replaced by a binary tree of zero-length auxiliary edges. This binarization preserves all leaf-to-leaf distances and produces a binary tree with $O(k)$ nodes. Contraction and binarization take time linear in the size of the constructed hierarchy.

Once the binary tree has been formed, the subtree masses $h_D(v)$ are computed from the leaf histogram by a single bottom-up traversal in $O(k)$ time and $O(k)$ space.

Dynamic programming and sampling. For the direct implementation of Algorithm 1, each internal node computes $Z_v(t) = \psi_v(t) \sum_{s=0}^{t} Z_u(s) Z_z(t-s)$ for every $t = 0, \dots, m$. Since the binary tree has $O(k)$ nodes, this requires $O(km^2)$ arithmetic operations. The top-down sampling phase evaluates at most $m+1$ possible splits at each internal node and therefore takes $O(km)$ additional time. Writing the final dataset requires $O(m)$ time.

Using the $O(k^2)$ FRT implementation described above, the total runtime of Algorithm 1 is $O\left(nk + k^2 + km^2\right)$, where the three terms correspond respectively to nearest-neighbor assignment, FRT construction, and exponential-mechanism sampling.

Storing a dense candidate distance matrix requires $O(k^2)$ space. The compressed binary tree, its edge weights, and the subtree masses require $O(k)$ space. Storing all partition values $Z_v(t)$ needed by the top-down sampler requires $O(km)$ space. Thus the overall space complexity is $O(k^2 + km)$.

C Details on hierarchical tree-projection theorem

We detail the procedure that we described in the main body in Algorithm 2.

Next, we prove Theorem 2. We start considering a fixed public FRT hierarchy T and then average over the FRT randomness. All expectations in the fixed-tree part are over the Laplace noises only. Recall $\mathcal{L} = \{\ell : K_\ell \geq 2\}$. For $v \in \mathcal{P}_\ell$, write $h_\ell(v) := h_D(v)$, $q_{a,\ell}(v) := a(v)/m$, and $\xi_v := Z_v$. Since the incoming edge of every node in $\mathcal{P}_\ell$ has length w_ℓ, the tree-Wasserstein identity gives, for every $a \in \mathcal{A}_m$,

$$F(a) := W_{1,d_T}(h_D, q_a) = \sum_{\ell \in \mathcal{L}} w_\ell \sum_{v \in \mathcal{P}_\ell} |h_\ell(v) - q_{a,\ell}(v)|. \tag{10}$$

Indeed, if $K_\ell = 1$, then $\mathcal{P}_\ell = \{C\}$ and both measures assign mass one to its unique cell, so that level contributes zero. The objective in Equation (4) can be written

$$\widehat{F}(a) = \sum_{\ell \in \mathcal{L}} w_\ell \sum_{v \in \mathcal{P}_\ell} |h_\ell(v) + \xi_v - q_{a,\ell}(v)|. \tag{11}$$

Privacy. Fix $\ell \in \mathcal{L}$. Under replacement adjacency, the histogram vector $(h_D(v))_{v \in \mathcal{P}_\ell}$ changes by either zero or by $(e_u - e_v)/n$ for two cells $u, v \in \mathcal{P}_\ell$. Hence its ℓ_1 sensitivity is at most $2/n$. Releasing all coordinates with independent $\mathrm{Lap}(2/(n\varepsilon_\ell))$ noise is therefore ε_ℓ-DP. The allocation ε_ℓ in Algorithm 2 depends only on the public tree and on k, m, ε, and satisfies $\sum_{\ell \in \mathcal{L}} \varepsilon_\ell = \varepsilon$. Sequential composition over levels therefore makes the collection $\{y_v : v \in \mathcal{P}_\ell, \ell \in \mathcal{L}\}$ ε-DP. Every subsequent operation in Algorithm 2 is deterministic postprocessing of these releases and public information, so the output is ε-DP.

A deterministic comparison inequality. Fix any $a^* \in \arg\min_{a \in \mathcal{A}_m} F(a)$ and let $\widehat{a}$ be the minimizer returned by Algorithm 2. For each $\ell \in \mathcal{L}$, define

$$S_\ell := \mathrm{supp}(q_{a^*,\ell}) \cup \mathrm{supp}(q_{\widehat{a},\ell}).$$

At any level, a size-m multiset has positive mass in at most m cells. Thus

$$|S_\ell| \leq R_\ell := \min\{K_\ell, 2m\}. \tag{12}$$

For $v \notin S_\ell$, both $q_{a^*,\ell}(v)$ and $q_{\widehat{a},\ell}(v)$ are zero. Hence the corresponding term $w_\ell |h_\ell(v) + \xi_v|$ is identical in $\widehat{F}(a^*)$ and $\widehat{F}(\widehat{a})$ and can be subtracted from the optimality inequality $\widehat{F}(\widehat{a}) \leq \widehat{F}(a^*)$. Therefore

$$\sum_{\ell \in \mathcal{L}} w_\ell \sum_{v \in S_\ell} |h_\ell(v) + \xi_v - q_{\widehat{a},\ell}(v)| \leq \sum_{\ell \in \mathcal{L}} w_\ell \sum_{v \in S_\ell} |h_\ell(v) + \xi_v - q_{a^*,\ell}(v)|. \tag{13}$$

Algorithm 2 Hierarchical histogram with tree projection

Require: Private dataset D, public raw FRT hierarchy T, output size m, privacy parameter ε

Compute the NN histogram h_D at the leaves and, by a bottom-up traversal of T, the subtree masses $h_D(v)$ for every original node v.
Let $\mathcal{L} \leftarrow \{\ell \in \{1, \dots, L\} : K_\ell \geq 2\}$.
for each $\ell \in \mathcal{L}$ **do**
    Set $R_\ell \leftarrow \min\{K_\ell, 2m\}$ and $B_\ell \leftarrow w_\ell R_\ell \log(eK_\ell/R_\ell)$.
end for
Set $\varepsilon_\ell \leftarrow \varepsilon\sqrt{B_\ell}/\big(\sum_{j\in\mathcal{L}} \sqrt{B_j}\big)$ for every $\ell \in \mathcal{L}$.
for each $\ell \in \mathcal{L}$ and each $v \in \mathcal{P}_\ell$ **do**
    Draw $Z_v \sim \mathrm{Lap}(2/(n\varepsilon_\ell))$ independently and set $y_v \leftarrow h_D(v) + Z_v$.
end for
Binarize every node with more than two children using zero-weight auxiliary nodes.
Define $\ell_v(t)$ for every node and $t = 0, \dots, m$ by Equation (5).
for each node v in bottom-up order **do**
    if v is a leaf **then**
        $\widehat{F}_v(t) \leftarrow \ell_v(t)$ for every $t = 0, \dots, m$.
    else if v has one child u **then**
        $\widehat{F}_v(t) \leftarrow \ell_v(t) + \widehat{F}_u(t)$ for every $t = 0, \dots, m$.
    else
        Let u, z be the two children of v.
        for $t = 0, \dots, m$ **do**
            $s_v(t) \leftarrow \min\Big(\arg\min_{s\in\{0,\dots,t\}}\{\widehat{F}_u(s) + \widehat{F}_z(t-s)\}\Big)$.
            $\widehat{F}_v(t) \leftarrow \ell_v(t) + \widehat{F}_u(s_v(t)) + \widehat{F}_z(t - s_v(t))$.
        end for
    end if
end for
Set $t_r \leftarrow m$.
Traverse the tree top-down: at a unary node v with child u, set $t_u \leftarrow t_v$; at a binary node v with children u, z, set $t_u \leftarrow s_v(t_v)$ and $t_z \leftarrow t_v - s_v(t_v)$.
For each candidate leaf c_j, set $\widehat{a}_j \leftarrow t_{c_j}$.
return $\widehat{a}_j$ copies of each candidate c_j.

The terms of the true objective outside S_ℓ are likewise identical for a^* and $\widehat{a}$. Using $|h - q| \leq |h + \xi - q| + |\xi|$ on the left of (13) and $|h + \xi - q| \leq |h - q| + |\xi|$ on the right gives

$$F(\widehat{a}) \leq F(a^*) + 2\sum_{\ell\in\mathcal{L}} w_\ell \sum_{v\in S_\ell} |\xi_v|. \tag{14}$$

No independence is used here; in particular, (14) remains valid although the random set S_ℓ depends on all released noises.

For each level, let $|Z^{(\ell)}|_{(1)} \geq \cdots \geq |Z^{(\ell)}|_{(K_\ell)}$ be the decreasing order statistics of $\{|Z_v| : v \in \mathcal{P}_\ell\}$. By (12), deterministically $\sum_{v\in S_\ell} |Z_v| \leq \sum_{j=1}^{R_\ell} |Z^{(\ell)}|_{(j)}$. Hence

$$F(\widehat{a}) \leq F(a^*) + 2\sum_{\ell\in\mathcal{L}} w_\ell \sum_{j=1}^{R_\ell} |Z^{(\ell)}|_{(j)}. \tag{15}$$

Laplace order statistics and privacy allocation. If $Z_1, \dots, Z_K$ are iid $\mathrm{Lap}(2/(n\eta))$, then $|Z_i|$ are iid exponential random variables with mean $b = 2/(n\eta)$. Let $E_{(1)} \geq \cdots \geq E_{(K)}$ denote their decreasing order statistics. The standard exponential-spacing identity gives $\mathbb{E}E_{(j)} = b(H_K - H_{j-1})$.

Therefore, for $1 \le R \le K$,

$$\mathbb{E}\sum_{j=1}^{R} E_{(j)} = b\sum_{j=1}^{R}(H_K - H_{j-1}) \tag{16}$$

$$= bR(1 + H_K - H_R) \tag{17}$$

$$\le bR\log\frac{eK}{R}. \tag{18}$$

Taking conditional expectations in (15) and applying this bound independently at each level yields

$$\mathbb{E}[F(\widehat{a}) \mid T] \le F(a^*) + \frac{4}{n}\sum_{\ell\in\mathcal{L}}\frac{B_\ell}{\varepsilon_\ell}, \qquad B_\ell := w_\ell R_\ell \log\frac{eK_\ell}{R_\ell}. \tag{19}$$

All B_ℓ are strictly positive for $\ell \in \mathcal{L}$. By Cauchy–Schwarz,

$$\left(\sum_{\ell\in\mathcal{L}}\sqrt{B_\ell}\right)^2 \le \left(\sum_{\ell\in\mathcal{L}}\frac{B_\ell}{\varepsilon_\ell}\right)\left(\sum_{\ell\in\mathcal{L}}\varepsilon_\ell\right) = \varepsilon\sum_{\ell\in\mathcal{L}}\frac{B_\ell}{\varepsilon_\ell}. \tag{20}$$

Equality in (20) holds exactly for the allocation in Algorithm 2. Substituting that allocation into (19), and using $F(a^*) = \min_{a\in\mathcal{A}_m} F(a)$, proves that

$$\mathbb{E}[W_{1,d_T}(h_D, q_{\widehat{a}}) \mid T] \le \min_{a\in\mathcal{A}_m} W_{1,d_T}(h_D, q_a) + \frac{4}{n\varepsilon}\left(\sum_{\ell\in\mathcal{L}}\sqrt{w_\ell R_\ell \log\frac{eK_\ell}{R_\ell}}\right)^2. \tag{21}$$

Simplifying the levelwise term. Let $R := \min\{k, 2m\}$ and $r := \min\{k, m\}$. For every $\ell \in \mathcal{L}$,

$$R_\ell \log\frac{eK_\ell}{R_\ell} \le R\log\frac{ek}{R}. \tag{22}$$

To verify (22), first suppose $K_\ell \le R$. Then $R_\ell = K_\ell$, so the left side equals $K_\ell \le R \le R\log(ek/R)$. If $K_\ell > R$, then $R_\ell = R$ and the claim follows from $K_\ell \le k$.

Because the partitions refine with ℓ, the set $\mathcal{L}$ is of the form $\{\ell_0, \ell_0 + 1, \dots, L\}$ for its smallest element ℓ_0. By the raw FRT normalization, $w_{\ell_0+j} = 2^{-j} w_{\ell_0}$. Therefore

$$\sum_{\ell\in\mathcal{L}}\sqrt{w_\ell} \le \sqrt{w_{\ell_0}}\sum_{j=0}^{\infty} 2^{-j/2} = O(\sqrt{w_{\ell_0}}). \tag{23}$$

Since $K_{\ell_0} \ge 2$, choose leaves in two distinct cells of $\mathcal{P}_{\ell_0}$. Their path contains the two distinct incoming edges of those cells, each of length w_{ℓ_0}, and hence $D_T \ge 2w_{\ell_0}$. Combining (22) and (23) gives

$$\left(\sum_{\ell\in\mathcal{L}}\sqrt{B_\ell}\right)^2 \le R\log\frac{ek}{R}\left(\sum_{\ell\in\mathcal{L}}\sqrt{w_\ell}\right)^2 \tag{24}$$

$$\le O(D_T)R\log\frac{ek}{R}. \tag{25}$$

Finally, $R\log(ek/R) = O(r\log(ek/r))$: if $2m \le k$, then $R = 2m$ and $r = m$; if $2m > k$, then $R = k$ and $k/2 < r \le k$, so both quantities are $\Theta(k)$. Substitution into (21) proves (7).

Returning to the original metric. Fix $a_d^* \in \arg\min_{a\in\mathcal{A}_m} W_{1,d}(h_D, q_a)$, so $W_{1,d}(h_D, q_{a_d^*}) = \mathrm{OPT}_{d,m}(D)$. Since $d \le d_T$ pointwise on C, for every output a,

$$W_{1,d}(\mu_D, q_a) \le A(D,C) + W_{1,d}(h_D, q_a) \le A(D,C) + W_{1,d_T}(h_D, q_a).$$

Conditioning on T, applying (7), and comparing its minimum with a_d^* gives

$$\mathbb{E}[W_{1,d}(\mu_D, q_{\widehat{a}}) \mid T] \le A(D,C) + W_{1,d_T}(h_D, q_{a_d^*}) + O\left(\frac{D_T\, r\log(ek/r)}{n\varepsilon}\right). \tag{26}$$

The comparator a_d^* depends on D but not on the public FRT randomness. Applying the FRT stretch bound to a fixed optimal d-coupling between h_D and $q_{a_d^*}$ gives

$$\mathbb{E}_T W_{1,d_T}(h_D, q_{a_d^*}) \le O(\log k) W_{1,d}(h_D, q_{a_d^*}) = O(\log k)\,\mathrm{OPT}_{d,m}(D).$$

Moreover, the FRT normalization in Section A gives $D_T \le 4w_1 = O(\mathrm{diam}_d(C))$ for every sampled tree. Taking expectation over T in (26) proves (8). Since T is sampled independently of D, mixing the fixed-T ε-DP mechanisms over this public randomness also preserves ε-DP.

Exactness of the min-plus dynamic program. Consider the binarized tree used by Algorithm 2. For a node v and $t \in \{0, \dots, m\}$, let $\mathcal{A}_v(t)$ be the set of nonnegative integer multiplicities on candidate leaves in the subtree of v whose sum is t. For such a leaf allocation a, extend the notation $a(u)$ to every original or auxiliary node u by summing the multiplicities of descendant candidate leaves. Define

$$\widehat{L}_v(a) := \sum_{u \in \mathrm{subtree}(v)} \ell_u(a(u)),$$

where ℓ_u is defined by Equation (5). Because the root and all auxiliary or unmeasured nodes have zero local cost, for a global allocation $a \in \mathcal{A}_m$ we have $\widehat{L}_r(a) = \widehat{F}(a)$ exactly.

We prove by induction from the leaves upward that

$$\widehat{F}_v(t) = \min_{a \in \mathcal{A}_v(t)} \widehat{L}_v(a). \tag{27}$$

If v is a leaf, $\mathcal{A}_v(t)$ contains the unique allocation assigning multiplicity t to that leaf, so (27) is exactly $\widehat{F}_v(t) = \ell_v(t)$. If v is unary with child u, every allocation in $\mathcal{A}_v(t)$ places the same t records in the subtree of u, so $\widehat{F}_v(t) = \ell_v(t) + \widehat{F}_u(t)$. If v is binary with children u, z, every allocation in $\mathcal{A}_v(t)$ induces a unique split $s = a(u) \in \{0, \dots, t\}$ and $a(z) = t - s$. Conversely, any such split together with allocations in $\mathcal{A}_u(s)$ and $\mathcal{A}_z(t-s)$ defines an allocation in $\mathcal{A}_v(t)$. Conditioning on the split s, the minimum subtree cost is $\ell_v(t) + \widehat{F}_u(s) + \widehat{F}_z(t-s)$ by the induction hypothesis; minimizing over s gives exactly (6).

Applying (27) at the root yields $\widehat{F}_r(m) = \min_{a \in \mathcal{A}_m} \widehat{F}(a)$. During the bottom-up pass, Algorithm 2 stores the smallest minimizing split $s_v(t)$ at every binary state. In the top-down pass, a unary node passes its assigned count unchanged to its child, while a binary node assigned count t_v passes counts $s_v(t_v)$ and $t_v - s_v(t_v)$ to its two children. Induction down the tree shows that each visited child state attains the value used in its parent's minimizing recurrence. Therefore the leaf counts $\widehat{a}_j = t_{c_j}$ produced by the backtracking satisfy $\widehat{F}(\widehat{a}) = \widehat{F}_r(m)$ and are an exact minimizer of Equation (4).

Runtime and space. Computing the NN leaf histogram by exhaustive search costs $O(nk)$ distance evaluations. The dense public candidate distance matrix uses $O(k^2)$ time and space. The $O(k^2)$ FRT construction in Section B.3 can be run while recording the FRT level at which each compressed edge begins and ends. Expanding the resulting compressed tree by inserting the omitted unary nodes recovers the raw hierarchy required here and costs $O(M_T)$ additional time and space, where M_T is the number of nonroot original nodes in that raw hierarchy. Thus construction of the representation used by Algorithm 2 costs $O(k^2 + M_T)$ time after NN assignment.

The subtree masses $h_D(v)$ are computed from the leaf histogram in one bottom-up traversal in $O(M_T)$ time. Constructing all local cost tables costs $O(M_T m)$. Binarization adds only $O(k)$ zero-weight auxiliary nodes: if d_v is the out-degree of an original internal node, replacing a node of degree $d_v > 2$ uses $d_v - 2$ auxiliaries, and

$$\sum_{v : d_v \geq 2} (d_v - 1) = k - 1 \quad \Longrightarrow \quad \sum_v (d_v - 2)_+ \leq k - 2.$$

After separating out unary nodes, the number of binary nodes is $O(k)$. A unary update costs $O(m)$, while at a binary node computing all states in (6) costs $\sum_{t=0}^{m} O(t+1) = O(m^2)$. Hence the dynamic program costs $O(M_T m + k m^2)$ time. The top-down reconstruction is linear in the number of nodes and is absorbed by this bound. Since $m \geq 1$, the $O(M_T)$ construction and traversal terms are also absorbed, giving total time

$$O(nk + k^2 + M_T m + k m^2).$$

For storage, the dense distance matrix costs $O(k^2)$. The raw hierarchy, noisy masses and subtree masses cost $O(M_T)$, and the value table $\widehat{F}_v(0{:}m)$ at each original node costs $O(M_T m)$. The $O(k)$ auxiliary binary nodes and their value/backtracking tables cost $O(km)$, which is $O(M_T m)$ because the k candidate leaves are among the M_T nonroot original nodes. Thus the total space usage is $O(k^2 + M_T m)$. Finally, we note that $M_T \leq kL$. This completes the proof of Theorem 2.

D PROOF OF THE SUPPORT-ADAPTIVE NN HISTOGRAM LEMMA

Let $x = nh_D$, let $\widehat{x} = \Pi_2(x + Z)$ be the Euclidean projection onto the count simplex $\{u \geq 0 : \sum_i u_i = n\}$, and put $\widetilde{h} = \widehat{x}/n$. Euclidean simplex projection has the form $\widehat{x}_i = (x_i + Z_i - \theta)_+$, where θ is chosen so that $\sum_i (x_i + Z_i - \theta)_+ = n$. Write $u := \|Z\|_\infty$ and $g(t) := \sum_i (x_i + Z_i - t)_+$. Since $Z_i \leq u$, we have $g(u) \leq \sum_i x_i = n$. Since $Z_i \geq -u$, every $x_i + Z_i + u \geq x_i \geq 0$, so $g(-u) = \sum_i (x_i + Z_i + u) \geq n$. As g is continuous and nonincreasing and $g(\theta) = n$, one may choose the projection threshold with $\theta \in [-u, u]$.

We now show that

$$(x_i - \widehat{x}_i)_+ \leq \min\{x_i, 2u\} \tag{28}$$

for every coordinate. The bound by x_i is immediate. If $\widehat{x}_i > 0$, then $x_i - \widehat{x}_i = \theta - Z_i \leq |\theta| + |Z_i| \leq 2u$. If $\widehat{x}_i = 0$ and the deficit is positive, then $x_i + Z_i - \theta \leq 0$, so $x_i \leq \theta - Z_i \leq 2u$. This proves (28).

For two nonnegative vectors $x, \widehat{x}$ with the same total mass n,

$$\mathrm{TV}(x/n, \widehat{x}/n) = \frac{1}{n} \sum_i (x_i - \widehat{x}_i)_+.$$

If S indexes the s largest coordinates of h_D, summing the bound $2u$ from (28) over S and the trivial bound x_i over S^c gives

$$\mathrm{TV}(h_D, \widetilde{h}) \leq \tau_s(h_D) + \frac{2s}{n} \|Z\|_\infty.$$

Taking the infimum over s proves that

$$\mathrm{TV}(h_D, \widetilde{h}) \leq \inf_{1 \leq s \leq k} \left\{ \tau_s(h_D) + \frac{2s}{n} \|Z\|_\infty \right\}.$$

Since the inequality holds simultaneously for every deterministic s, taking expectations gives

$$\mathbb{E}\,\mathrm{TV}(h_D, \widetilde{h}) \leq \inf_{1 \leq s \leq k} \left\{ \tau_s(h_D) + \frac{2s}{n} \mathbb{E}\|Z\|_\infty \right\}.$$

For iid $Z_i \sim \mathrm{Lap}(2/\varepsilon)$, the variables $|Z_i|$ are iid exponentials with mean $2/\varepsilon$, so $\mathbb{E}\|Z\|_\infty = (2/\varepsilon) H_k$. Let $a^* \in \arg\min_{a \in \mathcal{A}_m} \mathrm{TV}(h_D, q_a)$. TV-optimality of $R_m(\widetilde{h})$ gives

$$\mathrm{TV}(\widetilde{h}, R_m(\widetilde{h})) \leq \mathrm{TV}(\widetilde{h}, q_{a^*}) \leq \mathrm{TV}(\widetilde{h}, h_D) + t_m(h_D).$$

A second triangle inequality therefore yields

$$\mathrm{TV}(h_D, R_m(\widetilde{h})) \leq t_m(h_D) + 2\,\mathrm{TV}(h_D, \widetilde{h}).$$

Finally, for distributions supported on C, $W_{1,d} \leq \mathrm{diam}_d(C)\,\mathrm{TV}$, and $W_{1,d}(\mu_D, h_D) = A(D, C)$. Taking expectations and substituting $\mathbb{E}\|Z\|_\infty = (2/\varepsilon) H_k$ proves Lemma 3.

E ADDITIONAL EXPERIMENTAL DETAILS

Histogram baselines. We compare against a differentially private histogram, which projects private records onto their nearest public candidates, perturbs the resulting count histogram, projects the noisy vector onto the probability simplex, and rounds/samples to obtain an integer multiplicity vector $a \in \mathcal{A}_m$. We evaluate 8 baseline configurations combining:

- **Noise Distribution:** Laplace noise $\mathrm{Lap}(2/\varepsilon)$ for ε-DP and Gaussian noise $\mathcal{N}(0, \sigma^2)$ with $\sigma = \frac{2\sqrt{\ln(1.25/\delta)}}{\varepsilon}$ for (ε, δ)-DP (with $\delta = 10^{-6}$).
- **Simplex Projection:** ℓ_1 and ℓ_2 simplex projection onto $\{z \geq 0 : \sum_{j=1}^k z_j = n\}$.
- **Quantization to m Points:** Multinomial sampling $\mathrm{Multinomial}(m, p)$ and deterministic largest-remainder rounding (allocating $\lfloor m \cdot p \rfloor$ counts and distributing remainders by largest fractional parts).

Candidate-pool scaling experiment. We use d as the Euclidean metric on $\mathbb{R}^2$ and set $n = 100$, $m = 30$, and $\varepsilon = 1$, and vary $k \in \{50, 100, 200, 500, 1000, 2000, 5000\}$. Private records are drawn independently and uniformly from the disk centered at $(0, 0)$ with radius 0.25. Public candidates are drawn independently from a Gaussian mixture with centers $(0, 0)$, $(0, 1)$, $(1, 0)$, and $(1, 1)$, respective weights $(0.4, 0.3, 0.2, 0.1)$, and component covariance $0.08^2 I_2$. For each value of k, we perform ten trials, drawing fresh private and public datasets in every trial. All methods use the same datasets within a trial, and the two tree methods share the same sampled public tree. Gaussian histogram baselines use $\delta = 10^{-5}$. We report the mean Euclidean Wasserstein error $W_{1,d}(\mu_D, q_a)$ across trials, with error bars given by the empirical standard deviation, computed with divisor ten, divided by $\sqrt{10}$.

Three-regime experiment. This experiment uses one fixed candidate set shared across all methods and all three input histograms. We take $k = 400$, $n = 200$, $m = 20$, $\varepsilon = 1$, and twenty trials. The candidates form four clusters centered on a circle of radius 8; each cluster contains 100 candidates with within-cluster standard deviation 0.32. The ambient dimension is 400.

Only the NN count vector changes across the three regimes. In the sparse regime, one candidate in each cluster receives 50 votes, so the support size is four. In the tree-EM regime, the first 50 candidates in every cluster receive one vote each, giving support size 200. In the tree-projection regime, 50 candidates in a single cluster receive four votes each, while the other three clusters receive no mass. For every trial, tree-EM and tree projection use the same public raw FRT hierarchy before their mechanism-specific preprocessing.

The means reported in Table 1 are 0.6398, 1.0294, and 1.2035 in the sparse regime; 1.7545, 0.8367, and 1.3430 in the diffuse-across-clusters regime; and 5.2981, 1.4912, and 0.8531 in the diffuse-within-one-cluster regime, for the best NN histogram, single-tree tree-EM, and tree projection respectively. For each regime, "best NN histogram" is the best fixed configuration among the Laplace/Gaussian, ℓ_1/ℓ_2 simplex-projection, and sampling/deterministic-discretization variants evaluated in that experiment.

F Further extensions

The basic mechanism uses one public tree and allows arbitrary candidate multiplicities. We now describe two extensions that reduce the variability of the tree approximation and impose constraints on the minimum number of distinct released candidates.

F.1 Multiple trees

For any fixed pair of measures, one FRT tree has expected distortion $O(\log k)$, but an individual draw can be much worse. Drawing several independent trees reduces this variability: by Markov's inequality, each tree has constant probability of achieving $O(\log k)$ distortion, so with probability at least $1 - 2^{-J}$, one of J trees does so (after adjusting the constant). The joint mechanism below can favor that better tree without a separate private tree-selection step.

Let $T_1, \dots, T_J$ be public trees sampled independently of D, let $L_j(D, a) = W_{1,d_{T_j}}(h_D, q_a)$, and put $D_{\max} = \max_j \operatorname{diam}_{d_{T_j}}(C)$. One can sample a tree index and an allocation jointly from

$$\Pr(\widehat{J} = j, \widehat{a} = a) = \frac{\exp[-\lambda L_j(D, a)]}{\sum_{\ell=1}^{J} \sum_{b \in \mathcal{A}_m} \exp[-\lambda L_\ell(D, b)]}, \qquad \lambda = \frac{\varepsilon n}{2 D_{\max}}.$$

This is an exact exponential mechanism on the augmented output space $\{1, \dots, J\} \times \mathcal{A}_m$. It is implemented by computing the root partition value $Z_r^{(j)}(m)$ for every tree, sampling $\widehat{J} = j$ with probability proportional to $Z_r^{(j)}(m)$, and then running the ordinary top-down sampler in T_j. Its sensitivity is at most $D_{\max}/n$, hence it is ε-DP, and

$$\mathbb{E} L_{\widehat{J}}(D, \widehat{a}) \le \min_{j,a} L_j(D, a) + \frac{2 D_{\max}}{n\varepsilon} \log(J N_m).$$

The runtime and storage are multiplied by J if all tables are retained; storage can remain that of one tree by computing root normalizers sequentially and recomputing the selected tree's table.

This construction chooses the best tree and allocation jointly in the exponential-mechanism sense. It samples from the same joint distribution because its marginal probability for tree j is

$$\Pr(\widehat{J} = j) = \frac{Z_r^{(j)}(m)}{\sum_{\ell=1}^{J} Z_r^{(\ell)}(m)},$$

and, conditional on $\widehat{J} = j$, the ordinary tree sampler selects a with probability $\exp[-\lambda L_j(D, a)]/Z_r^{(j)}(m)$. Multiplying these two probabilities cancels $Z_r^{(j)}(m)$ and recovers the desired joint law.

This joint mechanism is generally preferable to privately choosing a tree and then running a second exponential mechanism within the selected tree since it avoids privacy composition by treating the pair (j, a) as one output of a single exponential mechanism.

F.2 Constraints on the number of distinct selected candidates

The basic mechanism always releases exactly m records, but it may place all m copies on only a few distinct candidates. In some applications, we additionally require the released dataset to contain at least m' distinct candidates, where $1 \le m' \le \min\{m, k\}$. In terms of the multiplicity vector, the constrained output space is

$$\mathcal{A}_{m,m'} = \{a \in \mathcal{A}_m : |\operatorname{supp}(a)| \ge m'\}, \qquad |\operatorname{supp}(a)| = |\{j : a_j > 0\}|.$$

The original mechanism corresponds to $m' = 1$, while $m' = m$ enforces sampling without replacement.

To impose this constraint, the dynamic program can track both the total number of records and the number of distinct candidates used in each subtree. Let $R = \min\{m, k\}$, and define $Z_v(t, \rho)$ to be the sum of the exponential-mechanism weights of all allocations that place exactly t records on the leaves below v and assign positive multiplicity to exactly ρ of those leaves.

At a leaf v, the feasible states are

$$Z_v(0, 0) = \psi_v(0), \qquad Z_v(t, 1) = \psi_v(t), \quad t = 1, \dots, m,$$

and all other states have value zero. If an internal node v has children u and z, then

$$Z_v(t, \rho) = \psi_v(t) \sum_{s=0}^{t} \sum_{\eta=0}^{\rho} Z_u(s, \eta) Z_z(t - s, \rho - \eta).$$

Indeed, s records and η distinct candidates are used in the first child subtree, while the remaining $t - s$ records and $\rho - \eta$ distinct candidates are used in the second.

At the root, we first sample the support size $\rho \in \{m', \dots, R\}$ with probability

$$\Pr(|\operatorname{supp}(\widehat{a})| = \rho) = \frac{Z_{\text{root}}(m, \rho)}{\sum_{\eta=m'}^{R} Z_{\text{root}}(m, \eta)},$$

and then proceed down the tree. At a node v assigned state (t, ρ), we jointly sample (s, η) with probability proportional to

$$Z_u(s, \eta) Z_z(t - s, \rho - \eta),$$

and assign states (s, η) and $(t - s, \rho - \eta)$ to its two children.

The same subtree-induction argument as in theorem 1 shows that this procedure samples exactly from the exponential mechanism restricted to $\mathcal{A}_{m,m'}$. Since the support constraint is public and independent of D, the sensitivity remains D_T/n and the mechanism remains ε-DP. Its accuracy guarantees are the same as before, except that the optimum and the size of the output space are replaced by

$$L^*_{T,m'}(D) = \min_{a \in \mathcal{A}_{m,m'}} L_T(D, a)$$

and

$$N_{m,m'} = |\mathcal{A}_{m,m'}| = \sum_{\rho=m'}^{R} \binom{k}{\rho} \binom{m-1}{\rho-1},$$

respectively. In particular,

$$\mathbb{E}L_T(D, \widehat{a}) \leq L^*_{T,m'}(D) + \frac{2D_T}{n\varepsilon} \log N_{m,m'}.$$

Thus the support constraint may increase the nonprivate optimum, because fewer outputs are feasible, but it decreases the logarithmic privacy penalty because $N_{m,m'} \leq N_m$.

A direct implementation takes $O(km^2R^2)$ time and $O(kmR)$ space.