On Differential Private ℓ_1, ℓ_2 and ℓ_p^p Distance Queries

Erzhi Liu[*†] **Jerry Yao-Chieh Hu**[*†‡] **Alex Reneau**[‡] **Zhao Song**[§] **Han Liu**[†♮]

[†] Center for Foundation Models and Generative AI, Northwestern University, Evanston, IL 60208, USA
Department of Computer Science, Northwestern University, Evanston, IL 60208, USA
[‡] Ensemble AI, San Francisco, CA 94133, USA
[§] Simons Institute for the Theory of Computing, UC Berkeley, Berkeley, CA 94720, USA
[♮] Department of Statistics and Data Science, Northwestern University, Evanston, IL 60208, USA

erzhiliu@u.northwestern.edu, jhu@u.northwestern.edu
alex@ensemblecore.ai, magic.linuxkde@gmail.com, hanliu@northwestern.edu

Abstract

We introduce a refined differentially private (DP) data structure for kernel density estimation (KDE) with ℓ_1, ℓ_2 and ℓ_p^p distance kernels, where p is a positive integer. This new DP data structure offers not only improved privacy-utility tradeoff but also better query efficiency over prior results. Specifically, we study the mathematical problem: given a similarity function f (or DP KDE) and a private dataset $X \subset \mathbb{R}^d$, our goal is to preprocess X so that for any query $y \in \mathbb{R}^d$, we approximate $\sum_{x \in X} f(x, y)$ in a differentially private fashion. The best previous algorithm for $f(x, y) = \|x-y\|_1$ is the node-contaminated balanced binary tree by [Backurs, Lin, Mahabadi, Silwal, and Tarnawski, ICLR 2024]. Their algorithm requires $O(nd)$ space and time for preprocessing with $n = |X|$. For any query point, the query time is $\alpha^{-1} d \log^2 n$, with an multiplicative error guarantee of $(1+\alpha)$-approximation and an additive error guarantee of $O(\epsilon^{-1}\alpha^{-0.5} d^{1.5} R \log^{1.5} n)$. In this paper, we use the same space and pre-processing time, improve the best previous result [Backurs, Lin, Mahabadi, Silwal, and Tarnawski, ICLR 2024] in three aspects

- We reduce query time by $\alpha^{-1} \log n$ factor
- We improve the approximation ratio from $1 + \alpha$ to 1
- We reduce the error dependence by a factor of $\alpha^{-0.5}$

From a technical perspective, our method of constructing the search tree differs from previous work [Backurs, Lin, Mahabadi, Silwal, and Tarnawski, ICLR 2024]. In prior work, for each query, the answer is split into $\alpha^{-1} \log n$ numbers, each derived from the summation of $\log n$ values in interval tree countings. In contrast, we construct the tree differently, splitting the answer into $\log n$ numbers, where each is a smart combination of two distance values, two counting values, and y itself. We believe our tree structure may be of independent interest.

1 Introduction

We propose a refined differentially private (DP) data structure for DP kernel density estimation, offering improved privacy-utility tradeoff over prior results without compromising efficiency. Let $X \subset \mathbb{R}^d$ be a private dataset, and let $f(x, y) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a similarity function[2], such as a kernel or distance function, between a user query $y \in \mathbb{R}^d$ and a private data point $x \in X$.

[*]Equal contribution. Version: November 5, 2026. Future updates are on arXiv. This work was done during JH's internship at Ensemble AI.

[2]In this work, we use "kernel query" and "distance query" interchangeably.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Problem 1 (DP Kernel Density Estimation Query)**.** The DP Kernel Density Estimation (KDE) query problem aims for an algorithm outputting a *private* data structure $D_X : \mathbb{R}^d \to \mathbb{R}$, that approximates the map $y \mapsto \sum_{x \in X} f(x, y)$ in a DP fashion[a]. Especially, we require D_X to remain private with respect to X, regardless of the number of queries.

[a]For a given dataset X and a given query y, an KDE query computes an approximation to $\sum_{x \in X} f(x, y)$.

Problem 1 is well-studied [1–9] due to its broad applicability and its importance in productizing large foundation models [10, 11]. What makes this problem interesting is its generic abstraction of many practical DP challenges in modern machine learning. These include generating synthetic data similar to a private dataset [12–15], and selecting a similar public dataset for pre-training ML models [16–18]. Such problem becomes more prevalent in this era of large foundation models [10, 11]. Essentially, these problems involve computing the similarity between a private dataset (i.e., X) and a processed data point (i.e., query y), thus falling under Problem 1.

In this work, we focus on the ℓ_1 kernel[3] (i.e., $f(x, y) = \|x - y\|_1$). By far, the algorithm with the best privacy-utility tradeoff and query efficiency for Problem 1 is by Backurs, Lin, Mahabadi, Silwal, and Tarnawski [9]. They propose a DP data structure via a node-contaminated balanced binary tree for ℓ_1 kernel. To be concrete, we begin by defining the similarity error between two data structures and then present their results.

Definition 1.1 (Similarity Error between Two Data Structures)**.** For a fixed query, let A represent the value output by our private data structure, and let A' represent the true value. We say that A has an error of (M, Z) for $M \geq 1$ and $Z \geq 0$, if $\mathbb{E}[|A - A'|] \leq (M - 1)A' + Z$. This implies a relative error of $M - 1$ and an additive error of Z. The expectation is taken over the randomness used by the data structure.

Let $n := |X| \geq 1$ be the public size of the dataset, and let $R > 0$ be a public bound fixed independently of the private sample, with $X \subset [0, R]^d$. Let $X_{i,j}$ be the j-th coordinate of the i-th data point, and let $\alpha \in [0, 1]$ be a parameter of the data structure selected before calculation. In asymptotic bounds, we use $\log n$ as shorthand for $\max\{1, \log_2 n\}$. For ℓ_1 kernel, Backurs et al. [9] give the error bound $(1 + \alpha, \alpha^{-0.5}\epsilon^{-1}Rd^{1.5}\log^{1.5} n)$ through their DP data structure.

Compared to the results of [9], we provide a refined DP data structure and improves both privacy-utility tradeoff and query efficiency. Specifically, we not only improve the ℓ_1 error bound to $(1, \epsilon^{-1}Rd^{1.5}\log^{1.5} n)$, but also the query time from $O(\alpha^{-1}d\log^2 n)$ to $O(d \log n)$. This makes our new DP data structure the latest best algorithm for Problem 1:

Theorem 1.1 (Informal Version of Theorem 3.2)**.** Given a dataset $X \subset [0, R]^d$ with public $n = |X| \geq 1$ and $R > 0$, and $\epsilon \in (0, 1)$, there is an algorithm that uses $O(nd)$ space to build an ϵ-DP data structure under replace-one adjacency, which supports the following operations:

- **INIT**(X, ϵ)**.** It takes dataset X and privacy parameter ϵ as input and spends $O(nd)$ time to build a data-structure.
- **QUERY**$(y \in \mathbb{R}^d)$**.** For each fixed y, it spends $O(d \log n)$ time to output a scalar A such that

$$\mathbb{E}[|A - A'|] \leq O(\epsilon^{-1}Rd^{1.5}\log^{1.5} n).$$

Notably, our improvements stem from a key observation that — in the balanced binary tree data structure proposed by Backurs et al. [9], each query answer is split into $\alpha^{-1}\log n$ values, derived from the summation of $\log n$ values in interval tree counts. We deem that this splitting is not essential and can be achieved through simple and efficient preprocessing steps.

To this end, we introduce a novel data representation in the tree nodes, where each node stores the sum of distances from one point to multiple points. This allows us to split the answer into only $\log n$ values, each being a smart combination of two distance values, two count values, and y itself. This additional information enhances each query calculation, leading to an improved privacy-utility tradeoff (Theorem 3.2 and Lemma 3.4) and faster query time (Lemma 3.3).

Lastly, we extend our results to ℓ_2 through Gaussian projection, following [9], and to ℓ_p^p through binomial expansion. The Euclidean result uses a public bounding ball and gives relative error α

[3]We remark that ℓ_1 query distance is fundamental. Namely, it is easy to generalize the ℓ_1 results to a wide range of kernels and distance functions following the provably dimensional reduction recipe of [9]. See Appendix E and F for the guarantees and comparisons for ℓ_2 and ℓ_p^p kernels.

plus additive error $O(\epsilon^{-1}\alpha^{-1}R\log^{5/2} n)$ with probability 0.99 for each fixed query. The power extension covers every positive integer p, with explicit p-dependent error and complexity bounds and no multiplicative slack. For the precise query domains and comparisons, see Appendix E and F, particularly Tables 2 and 3.

Related Work. We defer related work discussion to Section B due to page limits.

Organization. Section 2 includes the preliminaries, summarizes the prior best DP data structure by [9], and provides a high-level overview of our main results. Section 3 presents our new data structure for DP ℓ_1 kernel estimation along with its theoretical analysis. Sections E and F extends the ℓ_1 results to ℓ_2 and ℓ_p^p kernels. Lastly, Section 4 concludes the paper. We defer proof-of-concept experiments to Section A due to space limits.

References	Approx. Ratio	Error	Query Time	Init time	Space
[9]	$1+\alpha$	$\alpha^{-0.5}\epsilon^{-1}Rd^{1.5}\log^{1.5} n$	$\alpha^{-1}d\log^2 n$	nd	nd
Theorem 1.1	1	$\epsilon^{-1}Rd^{1.5}\log^{1.5} n$	$d\log n$	nd	nd

Table 1: Comparison of $\|x-y\|_1$ Results with Best Known Algorithm by Backurs et al. [9]. We ignore the big-O notation in the table, for simplicity of presentation. We use n to denote the number of points in dataset. We use d to denote the dimension for each point in the dataset. We assume $X \subset [0,R]^d$ for a public bound R. We use the ϵ to denote ϵ-DP.

2 Preliminaries

Notation. For a positive integer n, we use $[n]$ to denote $\{1,2,\cdots,n\}$. Datasets contain n records, which may repeat. For a vector x, we use $\|x\|_1 := \sum_{i=1}^n |x_i|$ to denote its ℓ_1-norm. We use $\mathbb{E}[\cdot]$ to denote the expectation. We use $\mathrm{Var}[\cdot]$ to denote the variance. We use $\Pr[\cdot]$ to denote the probability. We use $\mathsf{Laplace}(\lambda)$ random variable with parameter λ. It is known that $\mathbb{E}[\mathsf{Laplace}(\lambda)] = 0$ and $\mathrm{Var}[\mathsf{Laplace}(\lambda)] = 2\lambda^2$.

2.1 Differential Privacy

For completeness, we state the definitions of differential privacy [19, 20].

Definition 2.1 (Pure/Approximate Differential Privacy)**.** A randomized algorithm $M : \mathcal{X}^n \to \mathcal{Y}$ satisfies (ϵ, δ)-differential privacy if, for all $x, x' \in \mathcal{X}^n$ differing by replacement of one entry (with the dataset size n fixed and public) and for all events $E \subset \mathcal{Y}$, we have $\Pr[M(x) \in E] \leq e^\epsilon \cdot \Pr[M(x') \in E] + \delta$

For special case of $(\epsilon, 0)$-differential privacy, we use pure or pointwise ϵ-differential privacy to denote it. For the other case of $\delta > 0$, we denote it as approximate differential privacy.

We use basic composition to allocate the privacy budget across released statistics.

Lemma 2.1 (Basic Composition for Pure DP [20])**.** Let $M_i : \mathcal{X}^n \to \mathcal{Y}_i$ be ϵ_i-DP for $i \in [k]$, under the same adjacency relation. If the mechanisms use independent randomness, their joint release $(M_1(X), \ldots, M_k(X))$ is $(\sum_{i=1}^k \epsilon_i)$-DP.

2.2 [9]'s DP Data Structure: Node-Contaminated Balanced Tree

Backurs et al. [9] propose a data structure for general high-dimensional (e.g., d-dimensional) ℓ_1 kernel via 1-dimensional decomposition:

$$\sum_{x\in X} \|x-y\|_1 = \sum_{i=1}^d \sum_{x\in X} |x_i - y_i|.$$

Namely, they create a DP data structure for a d-dimensional dataset X by considering d copies 1-dimensional DP data structures (i.e., $\sum_{x\in X} |x_i - y_i|$ for $i \in [d]$). For each of these 1-dimensional DP data structures, they employ a node-contaminated balanced tree algorithm as follows.

Let $n = |X|$ be the size of the dataset X. Considering all input values are integer multiples of R/n in $[0, R]$,[4]

1. They use a classic balance binary tree of L layers in one dimension: a binary tree where each non-leaf node has exactly two children, and all leaf nodes are at the same depth, L. The tree has $2^L - 1$ total nodes, with 2^{L-1} leaf nodes and L levels from the root (level 1) to the leaves (level L). Each level k contains 2^{k-1} nodes.
2. They assign the leaves to correspond to these multiples (of R/n in $[0, R]$) and store the number of dataset points at each position, while internal nodes store the sum of their children.
3. It's well-known that any interval query can be answered by summing values from $O(\log n)$ tree nodes. This provide an efficient (*noise-less*) query operation.
4. To ensure DP, they propose a noisy version of the tree by injecting noise onto each node, i.e., a *node-contaminated* balanced tree. A replacement affects at most two nodes per level, each count by at most one, so the count vector has $O(\log n)$ sensitivity.

To highlight the significance of this work — removing the α dependence in the result and eliminating a $\log n$ factor in the query (see Table 1) — we make the following remarks on [9].

- **Query Time.** For each query y, in order to report the answer, they need to take summation of $O(\alpha^{-1} \log n)$ numbers where each number from each region $(y + (1+\alpha)^i, y + (1+\alpha)^{i+1}]$ (see line 7 in Algorithm 3 in [9]). In particular, for each region, they need to run a interval query which consists of $\log n$ numbers (each number is from a tree node, see Algorithm 2 in [9]). That's why the their algorithm has $\alpha^{-1} \log^2 n$ dependence.
- **Error Bound and Accuracy.** Due to the region $(y + (1+\alpha)^i, y + (1+\alpha)^{i+1}]$ can only approximate the number within $(1+\alpha)$ approximation. That is why the final error is proportional to $\alpha^{-1/2}$ and accuracy is losing a $(1+\alpha)$ relative error. Since the values are in range $[0, R]$, thus the error bound will be linearly depend on R according to how they construct the tree and alayze the error.
- **Sensitivity and DP Guarantee.** Since the tree is analyzing the counting problem, thus the sensitivity is $O(\log n)$ because change one point, will affect $O(\log n)$-levels stored counts.

2.3 High-Level Overview of Our DP Data Structure

We improve [9] by providing a refined balanced tree for DP 1-dimensional ℓ_1 distance query. Specifically, we introduce a new data representation on each node of the balanced binary tree as follows.

Let $n = |X| \geq 1$ be public, and let $x_k \in [0, R]$ for $k \in [n]$, where $R > 0$ is public and fixed independently of X. (Note that unlike [9], we don't assume x_k is a multiple of R/n, because our improved algorithm is compatible with floating-point values of x_k.)

1. We use a balanced binary tree where in each node stores both distance s and the counts c. Here, s denotes the pre-summation of certain distances, and c denotes the pre-counting.
2. We make a key observation (Lemma 3.1) that these values (s and c) are not depending on query y, while they are critical components of the final answer (the ℓ_1 kernel $\sum_{x \in X} \|x - y\|_1$). This motivates us to combine the statistics from $L = \lceil \log_2 n \rceil$ layers, using $s_{\text{right}} - s_{\text{left}} + y(c_{\text{left}} - c_{\text{right}})$.
3. To ensure DP, we need a noise version of the tree, due to s and c have different sensitivities, thus we need to add different levels of noise for s and c.

A few remarks are in order:

- **Query Time.** For each query $y \in (0, R)$, we sum the count and sum statistics over $L = O(\log n)$ sibling nodes and return $s_{\text{right}} - s_{\text{left}} + y(c_{\text{left}} - c_{\text{right}})$ (Lemma 3.1). For $y \notin (0, R)$, we use the public count n and the released root sum.
- **Error Bound and Accuracy.** Since our algorithm does not use the $(1+\alpha)^i$ region concept, we avoid the $\alpha^{-1/2}$ error. As a result, our error is purely additive, with no relative error.

[4] For general continuous data, they achieve this by rounding all data points to some integer multiples of R/n.

- **Sensitivity and DP Guarantee.** A replacement affects at most two root-to-leaf paths, each with $h = L + 1$ nodes. The sum and count vectors have sensitivities at most $2hR$ and $2h$, respectively. Previous work by Backurs et al. [9] only use y to determine which region to count points, but our algorithm uses y to construct the final answer. For interior queries, $0 < y < R$, which explains the R factor between the noise scales of s and c.

3 A Refined Differentially Private Data Structure

We propose a new data structure with ϵ-differential privacy that achieves $O(\epsilon^{-1} R \log^{1.5} n)$ additive error. Our method improves the balanced binary tree structure proposed by Backur, Lin, Mahabadi, Silawal, and Tarnawsk [9] in both DP guarantee and efficiency. Our data structure adapts a new data representation on each node that simplifies each query without losing privacy guarantees.

3.1 Key Observation and New Data Structure for One Dimensional ℓ_1 Distance Query

For simplicity, we consider the 1-dimensional distance query problem with ℓ_1 kernel distance. Note that any high-dimensional distance query problem can be reduced to this case via the 1-dimensional decomposition discussed in Section 2.2 or in [9].

When doing queries, we observe that

Lemma 3.1. For a collection of values $\{x_1, x_2, \cdots, x_n\} \subset \mathbb{R}$ and a value y, we define two sets

$$S_+ := \{k \in [n] \ : \ x_k > y\}$$
$$S_- := \{k \in [n] \ : \ x_k < y\}.$$

It holds

$$\sum_{k=1}^{n} |x_k - y| = \underbrace{(\sum_{k \in S_+} x_k)}_{s_{\text{right}}} - \underbrace{(\sum_{k \in S_-} x_k)}_{s_{\text{left}}} + y \cdot \underbrace{|S_-|}_{c_{\text{left}}} - y \cdot \underbrace{|S_+|}_{c_{\text{right}}}.$$

Proof.

$$\begin{aligned}
\sum_{k=1}^{n} |x_k - y| &= \sum_{k \in S_+} (x_k - y) + \sum_{k \in S_-} (y - x_k) \\
&= (\sum_{k \in S_+} x_k) - (\sum_{k \in S_-} x_k) + y \cdot (\sum_{k \in S_-} 1) - y \cdot (\sum_{k \in S_+} 1) \\
&= (\sum_{k \in S_+} x_k) - (\sum_{k \in S_-} x_k) + y \cdot |S_-| - y \cdot |S_+|.
\end{aligned}$$

This completes the proof. □

Lemma 3.1 provides a neat decomposition of the ℓ_1 distance query into four components. This motivates us to design a new DP data structure that pre-computes these terms and storing them in a balanced tree for possible algorithmic speedup. Surprisingly, this preprocessing not only accelerates query time but also improves the privacy-utility tradeoff. The following discussion illustrates this.

Finite-separable aggregation. The same principle applies when, separately on each side of y, a kernel has a finite representation $f(x, y) = \sum_{r=1}^{m} a_r(y)\phi_r(x)$, with possibly different coefficients and statistics on the two sides. Public, data-independent choices of ϕ_r and public bounds on $|\phi_r(x)|$ allow privacy calibration of their interval aggregates, with the privacy budget shared across all released statistics. A query retrieves $O(m \log n)$ noisy statistics and evaluates the coefficients $a_r(y)$; its error depends on those coefficients and the statistic bounds. A uniformly δ-accurate representation adds at most $n\delta$ approximation error, in addition to noise and the contribution of the omitted query-containing cell. Separability alone does not give the distance kernel's small leaf bias. Weighted statistics fit this

principle when the weights have public bounds and the sensitivity of the weighted record contributions is accounted for. Useful error guarantees for further kernels require this additional analysis.

To calculate each part efficiently, we define our data representation on a balanced binary tree as follows:

- **Dataset**: Given $X = \{x_k\}_{k=1}^n$ with $x_k \in [0, R]$, we use public $n \geq 1$ and $R > 0$.
- **Tree Structure**: Let $L := \lceil \log_2 n \rceil$ and $h := L + 1$. Levels $l = 0, \ldots, L$ contain 2^l nodes indexed by $j = 0, \ldots, 2^l - 1$. Empty leaves are allowed.
- **Node Representation**: Node (l, j) represents $I_{l,j} := [jR/2^l, (j+1)R/2^l)$, except that the final interval includes R. Its children are $(l+1, 2j)$ and $(l+1, 2j+1)$.

For $z \in [0, R]$, its node at level l has index $j_l(z) := \min\{2^l - 1, \lfloor 2^l z/R \rfloor\}$. This convention assigns internal boundaries to the interval on their right and includes both endpoints.

Each node $I_{l,j}$ stores two values: $c_{l,j}$ and $s_{l,j}$. Here:

- $c_{l,j} := |\{k \in [n] : x_k \in I_{l,j}\}|$ represents the count of data points in the interval.
- $s_{l,j} := \sum_{x_k \in I_{l,j}} x_k$ represents the sum of the data points' values in the interval.

After calculating each $c_{l,j}$ and $s_{l,j}$, we add independent noise drawn from $\mathsf{Laplace}(4h/\epsilon)$ to each c value and from $\mathsf{Laplace}(4hR/\epsilon)$ to each s value. The INIT algorithm (Algorithm 1) outlines the construction of our data structure.

In the QUERY algorithm (Algorithm 2), we sum the released s and c values over sibling nodes along the path to y. These nodes partition the points outside the leaf containing y into those on its left and right. Their aggregates approximate the four terms in Lemma 3.1; Lemma 3.5 accounts for the omitted leaf. For $y \leq 0$ or $y \geq R$, the public count and released root sum suffice.

Next, we prove the time complexity, differential privacy property and error of results of out algorithm respectively.

Algorithm 1 Building Binary Tree

data structure FASTERTREE ▷ Theorem 3.1
members
  $T := \{c, s\}$ and public n, R, L, h ▷ Released counts and sums
end members
procedure INIT(X, n, ϵ, R) ▷ Public $n = |X|$ and $R > 0$; $X \subset [0, R]$
  $L \leftarrow \lceil \log_2 n \rceil$, $h \leftarrow L + 1$; initialize all $c_{l,j}, s_{l,j}$ to zero
  for each x_k in X **do**
    $j \leftarrow j_L(x_k) = \min\{2^L - 1, \lfloor 2^L x_k/R \rfloor\}$
    $c_{L,j} \leftarrow c_{L,j} + 1$
    $s_{L,j} \leftarrow s_{L,j} + x_k$
  end for
  for each layer l from $L - 1$ down to 0 **do**
    for $j = 0, \ldots, 2^l - 1$ **do**
      $c_{l,j} \leftarrow c_{l+1,2j} + c_{l+1,2j+1}$
      $s_{l,j} \leftarrow s_{l+1,2j} + s_{l+1,2j+1}$
    end for
  end for
  Add independent noises drawn from $\mathsf{Laplace}(4hR/\epsilon)$ to each $s_{l,j}$
  Add independent noises drawn from $\mathsf{Laplace}(4h/\epsilon)$ to each $c_{l,j}$
  $T \leftarrow \{c, s\}$
end procedure
end data structure

3.2 Time Complexity

Here we compute the init and query time of Algorithm 1 and 2, respectively.

Algorithm 2 One Dimensional Distance Query

data structure FASTERTREE ▷ Theorem 3.1
procedure QUERY($y \in \mathbb{R}$) ▷ Lemma 3.3, Lemma 3.5
  if $y \leq 0$ **then**
    return $s_{0,0} - ny$
  else if $y \geq R$ **then**
    return $ny - s_{0,0}$
  end if
  $s_{\text{left}}, c_{\text{left}}, s_{\text{right}}, c_{\text{right}} \leftarrow 0$ ▷ As previously defined
  for each layer l from 1 to L **do**
    $j \leftarrow j_l(y) = \min\{2^l - 1, \lfloor 2^l y / R \rfloor\}$
    if $j \mod 2 = 1$ **then** ▷ if node j is the right child of its parent
      $c_{\text{left}} \leftarrow c_{\text{left}} + c_{l,j-1}$
      $s_{\text{left}} \leftarrow s_{\text{left}} + s_{l,j-1}$
    else ▷ if node j is the left child of its parent
      $c_{\text{right}} \leftarrow c_{\text{right}} + c_{l,j+1}$
      $s_{\text{right}} \leftarrow s_{\text{right}} + s_{l,j+1}$
    end if
  end for
  return $s_{\text{right}} - s_{\text{left}} + y \cdot c_{\text{left}} - y \cdot c_{\text{right}}$ ▷ Lemma 3.1
end procedure
end data structure

Lemma 3.2 (Init Time)**.** With $L = \lceil \log_2 n \rceil$, INIT (Algorithm 1) uses $O(n)$ time and space.

Proof. There are $\sum_{l=0}^{L} 2^l = 2^{L+1} - 1 < 4n$ nodes, since $n \leq 2^L < 2n$. Finding a leaf index takes constant time. In INIT (Algorithm 1), we iterate over n data points, then iterate all nodes. Therefore the total time complexity of pre-processing is $O(n)$. □

Lemma 3.3 (Query Time)**.** With $L = \lceil \log_2 n \rceil$, QUERY (Algorithm 2) takes $O(\log n)$ time.

Proof. An interior query visits one sibling node at each of L levels. An exterior or endpoint query uses only the root sum. Both take $O(L + 1) = O(\log n)$ time. □

Remark 3.1 (Comparing with Prior Work)**.** For a dataset $X \subset \mathbb{R}^d$ of size $n = |X|$ and a parameter $\alpha \in [0, 1]$ selected before calculation, Lemma 3.3 improves the prior best query time in [9] by a factor of $\alpha^{-1} \log n$.

3.3 Privacy Guarantees

Here we provide our DP guarantee of our proposed DP data structure (i.e., Algorithm 1).

Lemma 3.4 (Differential Privacy)**.** For fixed public n and R, the data structure FASTERTREE returned by Algorithm 1 is ϵ-DP on $[0, R]^n$ under replace-one adjacency.

Proof. Let $F_c(X)$ and $F_s(X)$ be the vectors of noiseless counts and sums over all $2^{L+1} - 1$ nodes. Replacing one entry removes its contribution along one root-to-leaf path and adds the new contribution along another. Each path has h nodes.

Sensitivity for Storing Counts. The ℓ_1 sensitivity of F_c is at most $2h$. Independent Laplace noise of scale $4h/\epsilon$ therefore makes its release $\epsilon/2$-DP.

Sensitivity for Storing Sums. Each removed or added value lies in $[0, R]$, so the ℓ_1 sensitivity of F_s is at most $2hR$. Independent Laplace noise of scale $4hR/\epsilon$ makes its release $\epsilon/2$-DP.

Lemma 2.1 gives ϵ-DP for the joint release. All queries use only these released values and public parameters, so post-processing preserves privacy for any number of queries. □

3.4 Error Guarantee

Here we provide the (data-structure) similarity error of our proposed DP data structure.

Lemma 3.5 (Error Guarantee)**.** Let $X = \{x_k\}_{k=1}^n \subset [0, R]$, with public $n \geq 1$ and $R > 0$, and let $\epsilon \in (0, 1)$. For each fixed $y \in \mathbb{R}$, let $A' := \sum_{k=1}^n |x_k - y|$ and let A be the output of Algorithm 2, with $L = \lceil \log_2 n \rceil$ and $h = L + 1$. Then $\mathbb{E}[|A - A'|] \leq R + 8Rh^{3/2}/\epsilon = O(\epsilon^{-1} R \log^{1.5} n)$.

Proof. Write $A - A' = N - B$, where B is the omitted leaf contribution and N is the centered noise.

Part I. For $0 < y < R$, the sibling nodes partition all points outside $I_{L,j_L(y)}$. Lemma 3.1 therefore gives

$$\begin{aligned} B &= \sum_{x_k \in I_{L,j_L(y)}} |x_k - y| && \text{(Omitted leaf contribution)} \\ &\leq \frac{nR}{2^L} && \text{(Leaf width is } R/2^L\text{)} \\ &\leq R. && \text{(}2^L \geq n\text{)} \end{aligned}$$

For $y \leq 0$ or $y \geq R$, the root-sum formula is exact before adding noise, so $B = 0$.

Part II. For an interior query, each of the L visited sibling nodes contributes independent sum and count noises with scales $4hR/\epsilon$ and $4h/\epsilon$. Their signs do not affect the variance. Thus $\mathbb{E}[N] = 0$ and

$$\begin{aligned} \mathrm{Var}[N] &= 2L \left(\frac{4hR}{\epsilon}\right)^2 + 2Ly^2 \left(\frac{4h}{\epsilon}\right)^2 && \text{(Independent Laplace noises)} \\ &\leq \frac{64R^2h^3}{\epsilon^2}. && \text{(}|y| \leq R \text{ and } L \leq h\text{)} \end{aligned}$$

For an exterior or endpoint query, N is the signed root-sum noise. Its variance is $32h^2R^2/\epsilon^2$, which obeys the same bound. Consequently,

$$\begin{aligned} \mathbb{E}[|A - A'|] &\leq B + \mathbb{E}[|N|] && \text{(Triangle inequality)} \\ &\leq R + \sqrt{\mathrm{Var}[N]} && \text{(Fact D.1 and } B \leq R\text{)} \\ &\leq R + \frac{8Rh^{3/2}}{\epsilon} && \text{(Variance bound)} \\ &= O(\epsilon^{-1} R \log^{1.5} n). && \text{(}\epsilon < 1 \text{ and } h = O(\log n)\text{)} \end{aligned}$$

□

Remark 3.2 (Comparing with Prior Work)**.** For a dataset $X \subset \mathbb{R}^d$ of size $n = |X|$ and a parameter $\alpha \in [0, 1]$ selected prior to calculation, Lemma 3.5 improves the approximation ratio from $1 + \alpha$ to 1 and reduces the error dependence by a factor of $\alpha^{-0.5}$ compared to [9].

3.5 One Dimensional Differentially Private Data Structure

Collectively, we present the init time, space, query time, approximation error, and privacy guarantees for the 1-dimensional DP data structure (Algorithm 1, 2) in the next theorem. As a reminder, we address the 1-dimensional distance query problem (Problem 1 with $d = 1$) using the ℓ_1 kernel distance.

Theorem 3.1 (1-Dimensional ℓ_1 DP Distance Query)**.** Given $X \subset [0, R]$ with public $n = |X| \geq 1$ and $R > 0$, and $\epsilon \in (0, 1)$, there is an algorithm that uses $O(n)$ space to build a data structure (Algorithm 1, 2) which supports the following operations

- **INIT**(X, ϵ)**.** It takes dataset X and privacy parameter ϵ as input and spends $O(n)$ time to build a data-structure.
- **QUERY**$(y \in \mathbb{R})$**.** For each fixed y, it spends $O(\log n)$ time to output a scalar A such that

$$\mathbb{E}[|A - A'|] \leq O(\epsilon^{-1} R \log^{1.5} n).$$

Furthermore, the data structure is ϵ-DP.

Proof. By Lemma 3.2, we prove the storage space, and the initialization time. By Lemma 3.3, we prove the query time of the data-structure. By Lemma 3.4, we prove the differential privacy guarantee. By Lemma 3.5, we prove the error guarantee after adding the noise. □

3.6 High Dimensional ℓ_1 Distance Query

We now generalize our 1-dimensional DP data structure to d-dimensional DP distance queries. Specifically, we apply our 1-dimensional DP data structure independently to each dimension to compute high-dimensional queries.

Algorithm 3 High Dimensional Distance Query

Input: n points in the public box $[0, R]^d$, public n, R, privacy parameter ϵ, query $y \in \mathbb{R}^d$.
data structure HIGHDIMFASTERTREE
members
  T_i for $i \in [d]$ ▷ T_i represents a FasterTree in Algorithm 1 for $i-$th dimension.
end memebers
procedure INIT(X)
  Independently initialize T_i.INIT($\{x_{k,i}\}_{k=1}^n, n, \epsilon/d, R$) for $i \in [d]$ using Algorithm 1.
end procedure
procedure QUERY(y)
  Return the sum of the outputs of Algorithm 2 from T_i with query y_i for $i \in [d]$.
end procedure
end data structure

Lemma 3.6 (Differential Privacy)**.** The data structure HIGHDIMFASTERTREE returned by HIGHDIMFASTERTREE.INIT (Algorithm 3) is ϵ-DP.

Proof. A replacement changes at most one entry in each coordinate dataset. Lemma 3.4 gives ϵ/d-DP for each release, and Lemma 2.1 gives ϵ-DP for their joint release. □

Lemma 3.7 (Error Guarantee)**.** Let $X = \{x_k\}_{k=1}^n \subset [0, R]^d$, with public $n \geq 1$ and $R > 0$, and let $\epsilon \in (0, 1)$. For each fixed $y \in \mathbb{R}^d$, let $A' := \sum_{k=1}^n \|x_k - y\|_1$ and let A be the output of Algorithm 3. With $h = \lceil \log_2 n \rceil + 1$, we have

$$\mathbb{E}[|A - A'|] \leq dR + \frac{8Rd^{3/2}h^{3/2}}{\epsilon} = O(\epsilon^{-1}Rd^{1.5}\log^{1.5} n).$$

Proof. Let A_i and A'_i denote the released and true answers in coordinate i. The proof of Lemma 3.5, with privacy budget ϵ/d, gives $A_i - A'_i = N_i - B_i$, where $0 \leq B_i \leq R$, $\mathbb{E}[N_i] = 0$, and $\mathrm{Var}[N_i] \leq 64d^2R^2h^3/\epsilon^2$. The noises N_i are independent across coordinates. Therefore,

$$\begin{aligned}
\mathbb{E}[|A - A'|] &= \mathbb{E}\left[\left|\sum_{i=1}^d (N_i - B_i)\right|\right] && \text{(Coordinate decomposition)} \\
&\leq \sum_{i=1}^d B_i + \sqrt{\sum_{i=1}^d \mathrm{Var}[N_i]} && \text{(Triangle inequality, independence, Fact D.1)} \\
&\leq dR + \frac{8Rd^{3/2}h^{3/2}}{\epsilon} && \text{(Bias and variance bounds)} \\
&= O(\epsilon^{-1}Rd^{1.5}\log^{1.5} n). && (d, h \geq 1 \text{ and } \epsilon < 1)
\end{aligned}$$

□

Theorem 3.2 (d-Dimensional ℓ_1 DP Distance Query, Formal version of Theorem 1.1)**.** Given $X \subset [0, R]^d$ with public $n = |X| \geq 1$ and $R > 0$, there is an algorithm that uses $O(nd)$ space to

build an ϵ-DP data structure (Algorithm 3) under replace-one adjacency, which supports the following operations

- **INIT**$(X, \epsilon \in (0,1))$**.** It takes dataset X and privacy parameter $\epsilon \in (0,1)$ as input and spends $O(nd)$ time to build a data structure.
- **QUERY**$(y \in \mathbb{R}^d)$**.** For each fixed y, it spends $O(d \log n)$ time to output a scalar $A \in \mathbb{R}$ such that $\mathbb{E}[|A - A'|] \leq O(\epsilon^{-1} R d^{1.5} \log^{1.5} n)$.

Proof. By Lemma 3.6, we prove the privacy guarantees. By Lemma 3.7, we prove the error guarantee of the QUERY. This completes the proof. □

4 Concluding Remarks

We present a refined data structure for differentially private kernel density estimation over the ℓ_1 kernel. Our new DP data structure significantly improves the state of the art in both query efficiency and approximation quality. Specifically, it enhances the best-known node-contaminated balanced binary tree method [9] in three aspects: faster queries (Lemmas 3.2 and 3.3), exact ratios (rather than approximate), and reduced error dependence (Lemma 3.5). Moreover, our approach achieves a strictly better privacy-utility tradeoff without incurring additional preprocessing overhead (Theorem 3.2). We observe lower mean absolute relative error at every tested ϵ and lower query latency at every tested database size in our one-dimensional synthetic experiments with baseline $\alpha = 0.01$ (Figures 2 and 3). We also extend our results to ℓ_2, and ℓ_p^p kernels in Sections E and F.

Numerical Results. We defer numerical comparisons to Section A due to page limits.

Acknowledgments and Disclosure of Funding

This work was completed during JH's internship at Ensemble AI. JH would also like to thank Maojiang Su, Mimi Gallagher, Sara Sanchez, T.Y. Ball, Dino Feng and Andrew Chen for enlightening discussions on related topics. Authors thank anomalous reviewers and chairs for constructive comments. All authors declare no competing interests.

JH is supported by Ensemble AI. The content is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

References

[1] Rob Hall, Alessandro Rinaldo, and Larry A. Wasserman. Differential privacy for functions and functional data. *J. Mach. Learn. Res.*, 14(1):703–727, 2013.

[2] Zhiyi Huang and Aaron Roth. Exploiting metric structure for efficient private query release. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 523–534. SIAM, 2014.

[3] Ziteng Wang, Chi Jin, Kai Fan, Jiaqi Zhang, Junliang Huang, Yiqiao Zhong, and Liwei Wang. Differentially private data releasing for smooth queries. *J. Mach. Learn. Res.*, 17:51:1–51:42, 2016.

[4] Francesco Aldà and Benjamin I. P. Rubinstein. The bernstein mechanism: Function release under differential privacy. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1705–1711. AAAI Press, 2017.

[5] Benjamin Coleman and Anshumali Shrivastava. A one-pass distributed and private sketch for kernel sums with applications to machine learning at scale. In *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 3252–3265. ACM, 2021.

[6] Amol Aggarwal and Josh Alman. Optimal-degree polynomial approximations for exponentials and gaussian kernel density estimation. In *Proceedings of the 37th Computational Complexity Conference*, pages 1–23, 2022.

[7] Lianke Qin, Aravind Reddy, Zhao Song, Zhaozhuo Xu, and Danyang Zhuo. Adaptive and dynamic multi-resolution hashing for pairwise summations. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 115–120. IEEE, 2022.

[8] Tal Wagner, Yonatan Naamad, and Nina Mishra. Fast private kernel density estimation via locality sensitive quantization. In *International Conference on Machine Learning (ICML)*, 2023.

[9] Arturs Backurs, Zinan Lin, Sepideh Mahabadi, Sandeep Silwal, and Jakub Tarnawski. Efficiently computing similarities to private datasets. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[10] Zinan Lin, Sivakanth Gopi, Janardhan Kulkarni, Harsha Nori, and Sergey Yekhanin. Differentially private synthetic data via foundation model apis 1: Images. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[11] Chulin Xie, Zinan Lin, Arturs Backurs, Sivakanth Gopi, Da Yu, Huseyin A. 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. *CoRR*, abs/2403.01749, 2024.

[12] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. Using gans for sharing networked time series data: Challenges, initial promise, and open questions. In *IMC '20: ACM Internet Measurement Conference, Virtual Event, USA, October 27-29, 2020*, pages 464–483. ACM, 2020.

[13] Xuechen Li, Florian Tramèr, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[14] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[15] Yucheng Yin, Zinan Lin, Minhao Jin, Giulia Fanti, and Vyas Sekar. Practical gan-based synthetic IP header trace generation using netshare. In Fernando Kuipers and Ariel Orda, editors, *SIGCOMM '22: ACM SIGCOMM 2022 Conference, Amsterdam, The Netherlands, August 22 - 26, 2022*, pages 458–472. ACM, 2022.

[16] Charlie Hou, Hongyuan Zhan, Akshat Shrivastava, Sid Wang, Aleksandr Livshits, Giulia Fanti, and Daniel Lazar. Privately customizing prefinetuning to better match user data in federated learning. *CoRR*, abs/2302.09042, 2023.

[17] Da Yu, Sivakanth Gopi, Janardhan Kulkarni, Zinan Lin, Saurabh Naik, Tomasz Lukasz Religa, Jian Yin, and Huishuai Zhang. Selective pre-training for private fine-tuning. *Trans. Mach. Learn. Res.*, 2024, 2024.

[18] Xiang Yue, Huseyin A. Inan, Xuechen Li, Girish Kumar, Julia McAnallen, Hoda Shajari, Huan Sun, David Levitan, and Robert Sim. Synthetic text generation with differential privacy: A simple and practical recipe. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

[19] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[20] Cynthia Dwork, Guy N. Rothblum, and Salil Vadhan. Boosting and differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 51–60, 2010.

[21] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies, Volume 1 (Long and Short Papers)*, 2018.

[23] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *The Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[24] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[25] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[26] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4195–4205, 2023.

[27] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[28] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems (NeurIPS)*, 32, 2019.

[29] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[30] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[31] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, Lifang He, and Lichao Sun. Sora: A review on background, technology, limitations, and opportunities of large vision models, 2024.

[32] Xiangxin Zhou, Xiwei Cheng, Yuwei Yang, Yu Bao, Liang Wang, and Quanquan Gu. Decompopt: Controllable and decomposed diffusion models for structure-based molecular optimization. *arXiv preprint arXiv:2403.13829*, 2024.

[33] Xiangxin Zhou, Dongyu Xue, Ruizhe Chen, Zaixiang Zheng, Liang Wang, and Quanquan Gu. Antigen-specific antibody design via direct energy-based preference optimization. In *NeurIPS*, 2024.

[34] Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Diffusion language models are versatile protein learners. *arXiv preprint arXiv:2402.18567*, 2024.

[35] Lihao Wang, Yuning Shen, Yiqun Wang, Huizhuo Yuan, Yue Wu, and Quanquan Gu. Protein conformation generation via force-guided se (3) diffusion models. In *Forty-first International Conference on Machine Learning*, 2025.

[36] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *ACL*, 2024.

[37] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*, 2022.

[38] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021.

[39] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[40] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

[41] Xiangyu Qi, Yangsibo Huang, Yi Zeng, Edoardo Debenedetti, Jonas Geiping, Luxi He, Kaixuan Huang, Udari Madhushani, Vikash Sehwag, Weijia Shi, et al. Ai risk management should incorporate both safety and security. *arXiv preprint arXiv:2405.19524*, 2024.

[42] Lichao Sun, Yue Huang, Haoran Wang, Siyuan Wu, Qihui Zhang, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, et al. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*, 2024.

[43] Zhangheng Li, Junyuan Hong, Bo Li, and Zhangyang Wang. Shake to leak: Fine-tuning diffusion models can amplify the generative privacy risk. In *2024 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 18–32. IEEE, 2024.

[44] Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021.

[45] Nicolas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 5253–5270, 2023.

[46] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[47] OpenAI. Gpt-4 technical report. *ArXiv*, 2023.

[48] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[49] Anthropic. The claude 3 model family: Opus, sonnet, haiku, 2023. URL https://api.semanticscholar.org/CorpusID:268232499. Accessed: 2024-08-28.

[50] Jiahao Yu, Haozheng Luo, Jerry Yao-Chieh Hu, Wenbo Guo, Han Liu, and Xinyu Xing. Enhancing jailbreak attack against large language models through silent tokens. *arXiv preprint arXiv:2405.20653*, 2024.

[51] Haozheng Luo, Jiahao Yu, Wenxin Zhang, Jialong Li, Jerry Yao-Chieh Hu, Xingyu Xin, and Han Liu. Decoupled alignment for robust plug-and-play adaptation. *arXiv preprint arXiv:2406.01514*, 2024.

[52] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. "Do Anything Now": Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models. In *The ACM Conference on Computer and Communications Security (CCS)*, 2024.

[53] Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. Jailbreaker: Automated jailbreak across multiple large language model chatbots. *arXiv preprint arXiv:2307.08715*, 2023.

[54] Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, and Yang Liu. Jailbreaking chatgpt via prompt engineering: An empirical study. *arXiv preprint arXiv:2305.13860*, 2023.

[55] Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*, 2023.

[56] Rusheb Shah, Soroush Pour, Arush Tagade, Stephen Casper, Javier Rando, et al. Scalable and transferable black-box jailbreaks for language models via persona modulation. *arXiv preprint arXiv:2311.03348*, 2023.

[57] Jiahao Yu, Xingwei Lin, and Xinyu Xing. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. *arXiv preprint arXiv:2309.10253*, 2023.

[58] Yi Zeng, Yu Yang, Andy Zhou, Jeffrey Ziwei Tan, Yuheng Tu, Yifan Mai, Kevin Klyman, Minzhou Pan, Ruoxi Jia, Dawn Song, et al. Air-bench 2024: A safety benchmark based on risk categories from regulations and policies. *arXiv preprint arXiv:2407.17436*, 2024.

[59] Fengqing Jiang, Zhangchen Xu, Luyao Niu, Boxin Wang, Jinyuan Jia, Bo Li, and Radha Poovendran. Poster: Identifying and mitigating vulnerabilities in llm-integrated applications. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, pages 1949–1951, 2024.

[60] Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Madhushani Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, et al. Sorry-bench: Systematically evaluating large language model safety refusal behaviors. *arXiv preprint arXiv:2406.14598*, 2024.

[61] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 1322–1333. ACM, 2015.

[62] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In Nadia Heninger and Patrick Traynor, editors, *28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019*, pages 267–284. USENIX Association, 2019.

[63] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. Gan-leaks: A taxonomy of membership inference attacks against generative models. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 343–362. ACM, 2020.

[64] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 1897–1914. IEEE, 2022.

[65] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr, and Chiyuan Zhang. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[66] Niv Haim, Gal Vardi, Gilad Yehudai, Ohad Shamir, and Michal Irani. Reconstructing training data from trained neural networks. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[67] Florian Tramèr, Reza Shokri, Ayrton San Joaquin, Hoang Le, Matthew Jagielski, Sanghyun Hong, and Nicholas Carlini. Truth serum: Poisoning machine learning models to reveal their secrets. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pages 2779–2792. ACM, 2022.

[68] Da Yu, Huishuai Zhang, Wei Chen, and Tie-Yan Liu. Do not let privacy overbill utility: Gradient embedding perturbation for private learning. *ArXiv*, abs/2102.12677, 2021.

[69] Soham De, Leonard Berrada, Jamie Hayes, Samuel L. Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *ArXiv*, abs/2204.13650, 2022.

[70] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[71] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[72] Thomas Hofmann, Bernhard Schölkopf, and Alexander Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36, 01 2007.

[73] Mengdi Huai, Di Wang, Chenglin Miao, Jinhui Xu, and Aidong Zhang. Privacy-aware synthesizing for crowdsourced data. In *International Joint Conference on Artificial Intelligence*, 2019.

[74] Teddy Cunningham, Graham Cormode, and Hakan Ferhatosmanoğlu. Privacy-preserving synthetic location data in the real world. *17th International Symposium on Spatial and Temporal Databases*, 2021.

[75] Arturs Backurs, Moses Charikar, Piotr Indyk, and Paris Siminelakis. Efficient density evaluation for smooth kernels. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 615–626, 2018.

[76] Arturs Backurs, Piotr Indyk, and Tal Wagner. Space and time efficient kernel density estimation in high dimensions. *Advances in neural information processing systems*, 32, 2019.

[77] Josh Alman, Timothy Chu, Aaron Schild, and Zhao Song. Algorithms and hardness for linear algebra on geometric graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 541–552. IEEE, 2020.

[78] Moses Charikar, Mikhail Kapralov, Navid Nouri, and Paris Siminelakis. Kernel density estimation through density constrained near neighbor search. *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 172–183, 2020.

[79] Jiehao Liang, Zhao Song, Zhaozhuo Xu, and Danyang Zhuo. Dynamic maintenance of kernel density estimation data structure: From practice to theory. In *UAI*, 2025.

[80] Ainesh Bakshi, Piotr Indyk, Praneeth Kacham, Sandeep Silwal, and Samson Zhou. Subquadratic algorithms for kernel matrices via kernel density estimation, 2022. URL https://arxiv.org/abs/2212.00642.

[81] Baihe Huang, Zhao Song, Omri Weinstein, Junze Yin, Hengjie Zhang, and Ruizhe Zhang. A dynamic low-rank fast gaussian transform. *arXiv preprint arXiv:2202.12329*, 2022.

[82] Yichuan Deng, Wenyu Jin, Zhao Song, Xiaorui Sun, and Omri Weinstein. Dynamic kernel sparsifiers. *arXiv preprint arXiv:2211.14825*, 2022.

[83] Anupam Gupta, Aaron Roth, and Jonathan Ullman. Iterative constructions and private data release. In *Proceedings of the 9th International Conference on Theory of Cryptography*, TCC'12, page 339–356, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 9783642289132.

[84] Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to noninteractive database privacy. *J. ACM*, 60(2), may 2013. ISSN 0004-5411.

[85] Ardalan Mirshani, Matthew Reimherr, and Aleksandra Slavković. Formal privacy for functional data with gaussian perturbations. In *International Conference on Machine Learning*, pages 4595–4604. PMLR, 2019.

[86] Michael T Smith, Mauricio A Álvarez, Max Zwiessele, and Neil D Lawrence. Differentially private regression with gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 1195–1203. PMLR, 2018.

[87] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 715–724, 2010. doi: 10.1145/1806689.1806787.

[88] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. In *Proceedings of the 38th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 6755 of *Lecture Notes in Computer Science*, pages 405–417. Springer, 2011. doi: 10.1007/978-3-642-22012-8_32.

[89] Monika Henzinger, Jalaj Upadhyay, and Sarvagya Upadhyay. Almost tight error bounds on differentially private continual counting. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5003–5039. SIAM, 2023.

[90] Jirí Matousek. *Lectures on discrete geometry*, volume volume 212 of Graduate texts in mathematics. Springer, 2002.

[91] Guy E. Blelloch. Prefix sums and their applications. Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University, 1990. URL https://www.cs.cmu.edu/~guyb/papers/Ble93.pdf.

[92] David B. Kirk and Wen-mei W. Hwu. *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann, 3rd edition, 2016.

[93] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.

Appendix

A Proof-of-Concept Experiments **18**
A.1 Synthetic Data 18
A.2 Real-World Data 19
B Related Work **19**
C Limitations and Future Work **21**
D Basic Facts **21**
E ℓ_2 Distance Query **22**
E.1 Error and DP Guarantees 22
E.2 Init and Query Time 23
F ℓ_p^p Distance Query **24**
G Illustrations of Sum and Prefix Aggregation **28**

Impact Statement

Our results offer a combination of fast queries, tight accuracy, and robust privacy guarantees. These results are essential for large-scale data analysis scenarios where computational efficiency and data confidentiality are critical. By the formal nature of this work, we do not expect any immediate negative social impact.

LLM Usage Disclosure

We used large language models (LLMs) to aid and polish writing, such as improving clarity, grammar, and conciseness. We also used LLMs for retrieval and discovery, for example exhausting literature to identify potential missing related work. All technical content, proofs, experiments, and results are original contributions by the authors.

A Proof-of-Concept Experiments

We validate the efficacy of our theory in both synthetic and real-world experiments.

A.1 Synthetic Data

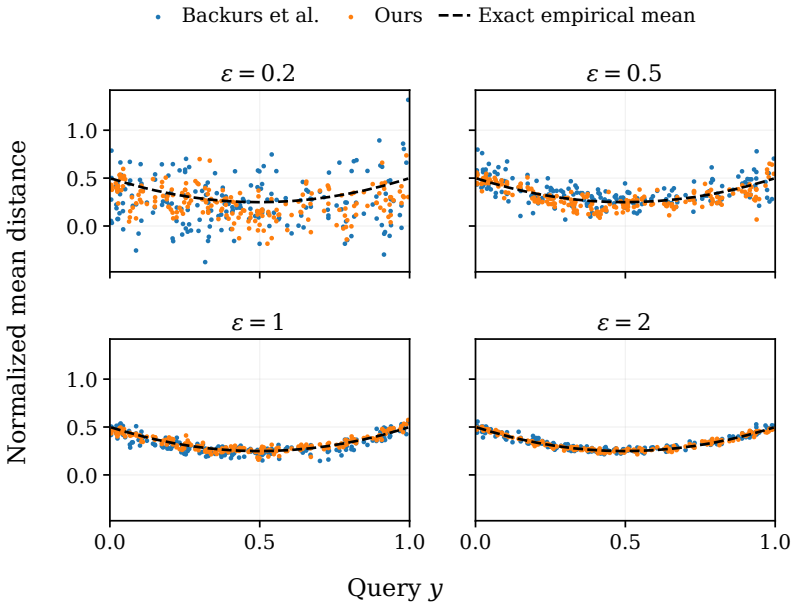


Figure 1: We compare individual releases from the first trial ($n = 10000$, 201 queries). The dashed line is the exact empirical mean.

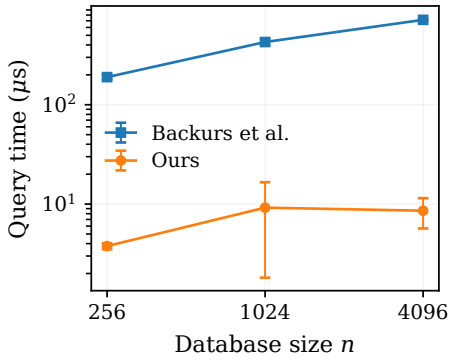


Figure 2: Mean query latency with pointwise 95% Student-t confidence intervals across 10 independent batches.

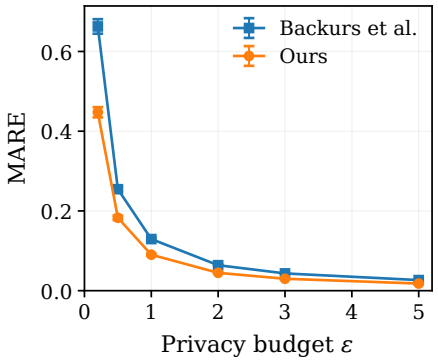


Figure 3: Mean absolute relative error with pointwise 95% Student-t confidence intervals across the 64 trial averages.

Here we present minimally sufficient numerical results to back up our theoretical analysis of the one-dimensional ℓ_1 algorithm.

Experiments Setup. In the ℓ_1 experiments, we evaluate the empirical mean distance $\bar{f}_X(y) = \frac{1}{n}\sum_{x \in X} |x - y|$ on the original data, where $n = |X|$ is public. We use the same data and queries for both methods, with public domain $[0, 1]$ and the same total privacy budget ϵ under fixed-size replace-one adjacency. For error measurements, we use 64 independent trials, each with $n = 10000$ records and 201 query points drawn independently and uniformly from $[0, 1]$.

Computational Resource. We conduct the synthetic experiments on an AMD EPYC 9V74 CPU using a single Python process. We implement both methods in Python.

Baseline. We choose [9] as our baseline and compare our algorithm with theirs on both running time and relative error. We set $\alpha = 0.01$ in their algorithm and use Algorithm 1 for our method.

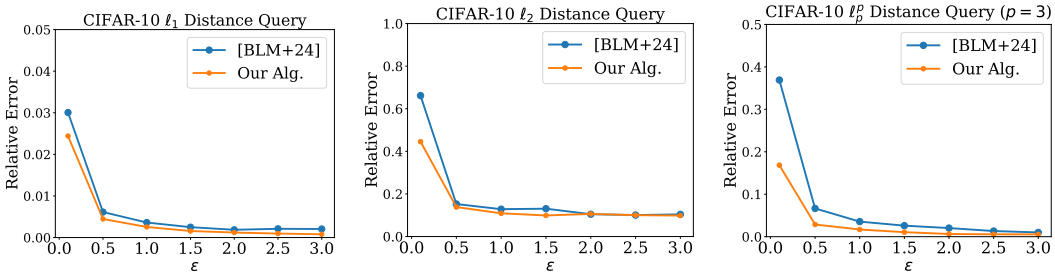
Approximation Test. In Figure 1, we compare the outputs of both algorithms with the exact empirical mean $\bar{f}_X$ at $\epsilon \in \{0.2, 0.5, 1, 2\}$. We show the first trial with $n = 10000$ and all 201 queries.

Running Time Test. In Figure 2, we compare the time per query for $n \in \{256, 1024, 4096\}$ at $\epsilon = 0.5$. We measure 10 independent batches of 201 queries using prepared data structures for each n and divide each batch time by 201, excluding initialization. We observe lower mean query latency for our algorithm at every tested n.

Relative Error Test. In Figure 3, we set $|X| = 10000$ and compute relative errors for $\epsilon \in \{0.2, 0.5, 1, 2, 3, 5\}$. We compute mean absolute relative error (MARE) by averaging the absolute relative errors across the 201 queries in each trial, then report its mean across the 64 trials. We observe lower mean MARE for our algorithm at every tested ϵ, with improvements of approximately 28.15%–32.88% over [9].

A.2 Real-World Data

Here we present our previously reported real-world experiments on CIFAR-10. Their generating code and run records are unavailable, so we cannot verify their higher-dimensional privacy calibration.



ℓ_1 queries ℓ_2 queries ℓ_p^p queries

Figure 4: Reported distance-query comparisons on CIFAR-10. The blue line is [9], and the orange line is ours.

Data. CIFAR-10 is a dataset consisting of 60000 32x32 color images. We project each of them into a real vector of dimension 100 and compute distance queries for various ϵ.

Experiments Setup. We compare our algorithm with [9].

In our experiments, we choose different $\epsilon \in \{0.1, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0\}$. Then we construct a database containing 10000 vectors of dimension 100 and each time we process 100 queries randomly drawn from the CIFAR-10 test batch.

Relative Error Test. We consider three types of distance queries: ℓ_1 (Figure 4(a)), ℓ_2 (Figure 4(b)) and ℓ_p^p with $p = 3$ (Figure 4(c)). For each of them, we report the average relative error of the queries. Figure 4 displays lower reported errors for our method on all three distance metrics.

B Related Work

Privacy, Security and Safety in Large Foundation Models. The motivating problems for this work come from the privacy concerns of large foundation models [10, 11]. In modern machine learning, Foundation models [21], including pretrained transformer and diffusion models, gain popularity in many AI applications due to their ability to generalize across diverse tasks with minimal fine-tuning. Pretrained transformers, such as BERT [22], GPT [23], and Llama [24, 25], leverage vast amounts of data to learn general-purpose, context-aware representations. Diffusion models [26–28], on the other

hand, excel in generative tasks, particularly in producing high-quality images and data distributions through iterative refinement [29–35]. Importantly, what makes them fundamental and versatile is their flexibility for diverse downstream tasks via fine-tuning methods [36–40], hence "foundation." Together, these models signify a shift towards more powerful AI systems that serve as the basis for a wide range of applications across different domains.

However, there exists a gap from productizing many these advancements due to privacy, safety, and security concerns [41–44]. These models, due to their vast scale and extensive training on large datasets, risk exposing sensitive information and amplifying biases present in the data. Privacy issues arise when models inadvertently memorize and reproduce personal data from training sets [45, 46]. Safety concerns include the potential for generating harmful or misleading content, especially in applications where accuracy is paramount [44, 47–57]. Security vulnerabilities also exist, as these models may be susceptible to adversarial attacks that manipulate outputs or extract proprietary information [58–60]. Addressing these challenges is essential to ensure the responsible and secure use of foundation models across various applications. In this work, we focus on the privacy of foundation models at a fundamental level by study the formalized KDE query Problem 1.

Differential Privacy (DP). Differential Privacy (DP) is a standard tool for understanding and mitigating privacy concerns in machine learning. Proposed by Dwork et al. [19], DP offers a robust approach to protecting sensitive information. Given the fact that non-private ML models expose sensitive user data [61–63, 46, 64–67, 45], researchers propose various methodologies. One approach involves generating synthetic datasets that closely resemble the original private data and training models on these synthetic datasets [12–15, 18, 10]. Another strategy is to use similar public examples for pre-training models [16, 17]. Additionally, the DP-SGD method has benefited from incorporating public data similar to private datasets to enhance downstream model performance [68, 69, 14, 15, 13, 16, 17, 10]. As highlighted by Backurs et al. [9], the common denominator of all these works is the need to *compute similarities to a private dataset*. In this work, we study this problem at a fundamental level by formalizing it as the DP-KDE query problem (Problem 1).

Kernel Density Estimation (KDE). Kernel Density Estimation (KDE) is a key technique in statistics and machine learning. This technique converts a collection of data points into a smoothed probability distribution [70–72]. It is prevalent in many private applications such as crowdsourcing and location sharing [73, 74]. Although research on non-private KDE already produce efficient methods [75–79, 7, 80–82], adapting these techniques to DP remains complex [83, 84, 4, 8, 9]. For the ℓ_1 distance-sum task, Backurs et al. [9] give the prior best-known bounds compared in Table 1. In this work, we improve these bounds in both privacy-utility trade-off and efficiency. Huang and Roth [2] study private average-distance queries over metric spaces, including offline release, using a reduction from online learning; multiplication by public n translates their average normalization to our sums. Wagner et al. [8] develop locality-sensitive quantization for private KDE, including efficient Gaussian-kernel mechanisms. Our direct quantitative comparison is with the distance-sum data structure of Backurs et al. [9], which shares our task and one-time-release setting, rather than with all private KDE mechanisms.

Comparison with Function-Sanitizing Approaches for Differentially Private KDE. Hall et al.(2013) [1] pioneer DP kernel density estimation by adding a Gaussian-process perturbation to the entire density function. This yields an (ϵ, δ)-DP sanitized function that can be queried anywhere, but it requires heavy noise and came with limited accuracy guarantees. Mirshani et al.(2019) [85] extend this "function release" approach using Gaussian process noise in Banach spaces, likewise achieving (ϵ, δ)-DP over the full function at a high privacy cost. Aldà and Rubinstein (2017) [4] instead propose the *Bernstein mechanism*, approximating the KDE with a finite polynomial basis and adding independent noise to each coefficient. This produces a published density function under (pure) DP with improved utility but requires truncating the function to a fixed basis. Smith et al.(2018) [86] apply DP to Gaussian process models of the KDE. They show that releasing an unrestricted GP-based density demands large noise, and they improve accuracy by assuming a fixed set of query points (reducing query flexibility). Like other function-release mechanisms, our tree is privatized once: all noisy node statistics are released, and subsequent queries only post-process them. Its distance-specific count and coordinate-sum representation yields the improved ℓ_1 bounds compared with Backurs et al. [9] in Table 1.

DP Tree and Matrix mechanisms. Dwork et al. [87] and Chan et al. [88] pioneer tree-based mechanisms for answering range queries under differential privacy. Their key insight is to store

counts in a balanced binary tree (or equivalently use the matrix mechanism) so that each query touches only $O(\log n)$ noisy aggregates, yielding lower error than naive solutions. Recent work by Henzinger et al. [89] further optimize these ideas to improve continual counting bounds. Our approach builds on this line of research. We also employ a balanced binary tree for fast, private query evaluation. However, we extend the classic tree mechanism by storing both counts and partial distance sums at each node, rather than just counts. This allows us to handle ℓ_1 kernel density estimation queries efficiently: we retrieve the relevant (noisy) partial sums from each level of the tree and combine them to compute $|x - y|$ over the dataset. Thus, while our algorithm inherits the core principle of aggregating noisy statistics along a tree, it generalizes the framework to distance-weighted queries, reducing error and removing the need for an additional approximation factor.

C Limitations and Future Work

Limitations. Our technique and analysis focus *only* on *static* datasets and the ℓ_1, ℓ_2, and ℓ_p^p kernels; extending the approach to other similarity measures or dynamic/streaming data remains an open challenge. For our ℓ_1 construction, space and preprocessing time match existing solutions at $O(nd)$ and may become restrictive in very high dimensions or for extremely large n. Moreover, although we reduce the query time by an $\alpha^{-1} \log n$ factor and improve approximation/error bounds, the constants involved in the data structure could be non-negligible in practice, especially under tight privacy constraints. Our $O(d \log n)$ query time is within a logarithmic factor of the cost of reading an explicit d-dimensional query. This does not establish query-time optimality: the lower bound of Backurs et al. [9] concerns additive error, and a matching query-time lower bound under comparable privacy, utility and space requirements is not established here.

Future Work. Our novel tree construction strategy may inspire further research on advanced data structures for privacy-preserving query processing. Future work could explore refining these constants, handling broader classes of kernels, and adapting the method to incremental or online data settings. We leave these directions for future exploration.

D Basic Facts

Fact D.1. Let X denote a random variable with $\mathbb{E}[X] = 0$. Then it holds

$$\mathbb{E}[|X|] \leq (\mathrm{Var}[X])^{1/2}.$$

Proof. It is easy to see that $(\mathbb{E}[|X|])^2 \leq \mathbb{E}[|X|^2] = \mathbb{E}[X^2]$. Next, we show

$$(\mathbb{E}[|X|])^2 \leq \mathbb{E}[X^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 \leq \mathrm{Var}[X].$$

Thus, we complete the proof. □

Fact D.2 (Union Bound)**.** Let $A_1, A_2, ..., A_n$ be a set of probability events. It holds

$$\Pr\left[\bigcup_{i=1}^{n} A_i\right] \leq \sum_{i=1}^{n} \Pr[A_i].$$

Proof. We prove union bound by induction. For $n = 1$, $\Pr[A_1] = \Pr[A_1]$. Suppose this inequality holds true for k, then for $k + 1$,

$$\begin{aligned}
\Pr\left[\bigcup_{i=1}^{k+1} A_i\right] &= \Pr\left[\bigcup_{i=1}^{k} A_i\right] + \Pr[A_{k+1}] - \Pr\left[(\bigcup_{i=1}^{k} A_i) \bigcap A_{k+1}\right] \\
&\leq \Pr\left[\bigcup_{i=1}^{k} A_i\right] + \Pr[A_{k+1}] \\
&\leq (\sum_{i=1}^{k} \Pr[A_i]) + \Pr[A_{k+1}]
\end{aligned}$$

$$= \sum_{i=1}^{k+1} \Pr[A_i].$$

Thus, we complete the proof. □

E ℓ_2 Distance Query

Using the Gaussian projection technique in [9], we extend our ℓ_1 result to the ℓ_2 norm. For a dataset in a public bounding ball, our construction gives additive error $O(\epsilon^{-1}\alpha^{-1}R\log^{5/2} n)$ with relative error α for each fixed query. The construction clips projected data to a public interval and accounts explicitly for the resulting distortion. Table 2 compares this bound with the reported $\widetilde{O}(\epsilon^{-1}\alpha^{-3/2}R)$ bound in [9].

References	Approx. Ratio	Error	Query Time	Init time	Space
[9], Cor. D.2	$1+\alpha$	$\widetilde{O}(\alpha^{-3/2}\epsilon^{-1}R)$	$k_0(d+\alpha^{-1}\log^2 n)$	ndk_0	$k_0(n+d)$
Theorem E.1	$1+\alpha$	$\alpha^{-1}\epsilon^{-1}R\log^{5/2} n$	$k(d+\log n)$	ndk	$k(n+d)$

Table 2: Bounds for ℓ_2 queries, with success probability 0.99 for each fixed query. Here $k = \Theta(\alpha^{-2}\log n)$ and $k_0 = \Theta(\alpha^{-2}\log n \log(2/\alpha))$. Big-O notation is suppressed except in the reported prior error, where $\widetilde{O}$ hides logarithmic factors in n and $1/\alpha$. The time and space entries count dense projection matrices and the corresponding ℓ_1 trees. Our result assumes a public ball $B_2(c, R)$; the prior corollary states a diameter bound.

E.1 Error and DP Guarantees

Lemma E.1 (Gaussian projection; [90])**.** Let $\gamma \in (0,1)$ and define $T : \mathbb{R}^d \to \mathbb{R}^k$ by

$$T(x)_i = \frac{1}{\beta k}\sum_{j=1}^{d} Z_{ij}x_j, \qquad i \in [k],$$

where $\beta = \sqrt{2/\pi}$ and the Z_{ij} are independent standard Gaussians. For every fixed $x \in \mathbb{R}^d$,

$$\Pr\left[(1-\gamma)\|x\|_2 \le \|T(x)\|_1 \le (1+\gamma)\|x\|_2\right] \ge 1 - 2e^{-\gamma^2 k/\pi}.$$

Proof. For $x \neq 0$, $\|T(x)\|_1/\|x\|_2$ has the same distribution as $(\beta k)^{-1}\sum_{i=1}^{k}|g_i|$, for independent standard Gaussians g_i. The sum has mean βk and is $\sqrt{k}$-Lipschitz in g. Gaussian concentration bounds its two-sided deviation by $2\exp\bigl(-t^2/(2k)\bigr)$. Taking $t = \gamma\beta k$ proves the claim; $x = 0$ is immediate. □

Theorem E.1. Let $n \ge 1$, $R > 0$, and $c \in \mathbb{R}^d$ be public parameters fixed independently of the private dataset $X \in B_2(c, R)^n$, where $B_2(c, R) := \{x : \|x - c\|_2 \le R\}$. For $\alpha, \epsilon \in (0,1)$, there is an ϵ-DP data structure under replace-one adjacency such that, for every fixed query $y \in \mathbb{R}^d$ independent of the mechanism's randomness, with probability at least 0.99 its output Z satisfies

$$\left|Z - \sum_{x\in X}\|x-y\|_2\right| \le \alpha\sum_{x\in X}\|x-y\|_2 + O(\epsilon^{-1}\alpha^{-1}R\log^{5/2} n).$$

The constant is absolute. Utility is pointwise in y; privacy holds for any number of queries.

Proof. We first construct the private release, then bound its projection, clipping, and query errors.

Construction and Privacy. Choose the public parameters

$$k := \lceil \pi\alpha^{-2} \ln(400n) \rceil, \qquad t := \sqrt{2\ln(2n)},$$
$$b := \frac{Rt}{\beta k}, \qquad h := \lceil \log_2 n \rceil + 1.$$

Sample T as in Lemma E.1. Write clip_b for coordinate-wise clipping to $[-b, b]$. Apply Algorithm 3, with total budget ϵ and public range $[0, 2b]^k$, to the records

$$u(x) := \mathrm{clip}_b(T(x - c)) + b\mathbf{1}.$$

Release T and the private trees. For a query y, use $v(y) := T(y - c) + b\mathbf{1}$ without clipping it. Algorithm 2 handles coordinates outside $[0, 2b]$ using its root formula. For every realized T, this fixed, record-wise transformation preserves replace-one adjacency. Lemma 3.6 therefore gives ϵ-DP conditionally on every T, and averaging over its data-independent distribution gives ϵ-DP for the joint release. This argument does not require any favorable clipping event. The public center locates the input domain; a diameter bound alone does not supply such a center.

Projection Error. Let $F := \sum_{x\in X} \|x - y\|_2$, $F_T := \sum_{x\in X} \|T(x - y)\|_1$, and $F_u := \sum_{x\in X} \|u(x) - v(y)\|_1$. Apply Lemma E.1 to each fixed difference $x - y$. A union bound gives

$$\begin{aligned} \Pr[|F_T - F| > \alpha F] &\le 2ne^{-\alpha^2 k/\pi} && \text{(Union bound over the } n \text{ differences)} \\ &\le \frac{1}{200}. && \text{(Choice of } k\text{)} \end{aligned}$$

Clipping and Private-Query Error. Define $D := \sum_{x\in X} \|T(x - c) - \mathrm{clip}_b(T(x - c))\|_1$. Translation cancels in F_u, so the reverse triangle inequality gives $|F_u - F_T| \le D$ for every y. Each projected data coordinate has standard deviation at most $R/(\beta k)$. For a standard Gaussian g, writing $(a)_+ := \max\{a, 0\}$, integration of its density gives $\mathbb{E}[(|g| - t)_+] \le \beta e^{-t^2/2}$. Consequently,

$$\begin{aligned} \mathbb{E}[D] &\le nk\frac{R}{\beta k}\mathbb{E}[(|g| - t)_+] && \text{(Bound each coordinate's standard deviation)} \\ &\le nRe^{-t^2/2} = \frac{R}{2}. && \text{(Gaussian tail integral and choice of } t\text{)} \end{aligned}$$

Conditioning on T fixes both $u(X)$ and $v(y)$. Lemma 3.7, with dimension k and range $2b$, yields

$$\mathbb{E}[|Z - F_u| \mid T] \le 2bk + \frac{16bk^{3/2}h^{3/2}}{\epsilon}.$$

The first term bounds the deterministic leaf bias; the second bounds the centered DP noise. Combining this with the clipping bound gives

$$\begin{aligned} \mathbb{E}[|Z - F_T|] &\le \frac{R}{2} + 2bk + \frac{16bk^{3/2}h^{3/2}}{\epsilon} =: M && \text{(Triangle inequality and total expectation)} \\ &= O(\epsilon^{-1}\alpha^{-1}R\log^{5/2} n). && \text{(Substitute } b, t, k, h \text{ and use } \epsilon < 1\text{)} \end{aligned}$$

Markov's inequality gives $|Z - F_T| \le 200M$ except with probability $1/200$. Combining the two failure probabilities and using $|Z - F| \le |Z - F_T| + |F_T - F|$ proves the theorem. Neither failure probability is a privacy parameter. □

E.2 Init and Query Time

We count real arithmetic operations and random samples at unit cost.

Init Time. Sampling and storing the dense projection takes $O(dk)$ operations. Mapping and clipping the n records takes $O(ndk)$ time, and constructing the k trees takes $O(nk)$ time. Thus initialization takes $O(\alpha^{-2}nd\log n)$ time. The released trees and projection use $O(k(n + d))$ space; the private records need not be retained.

Query Time. Mapping y takes $O(dk)$ time. The k one-dimensional queries take $O(k \log n)$ time, including endpoint and exterior queries. The total is $O(\alpha^{-2}(d + \log n) \log n)$.

F ℓ_p^p Distance Query

Suppose $p \geq 1$ is an integer. To calculate ℓ_p^p distance queries, we use binomial expansion techniques. Similar to ℓ_1, we first consider the one-dimensional case. We use public $n \geq 1$, $R > 0$, and integer $p \geq 1$, with $X \subset [0, R]$, $\epsilon \in (0, 1)$, and the interval convention from Section 3. Set $L := \lceil (\log_2 n)/p \rceil$, $h := L + 1$, and $m := \max\{1, L\}$. To display the dependence on p, define

$$K_p(n) := (p + 1)h\sqrt{m\binom{2p}{p}}.$$

All bounds below are uniform over positive integers p unless marked O_p.

References	Approx. Ratio	Error	Query Time	Init time	Space
[9], fixed p	$1 + \alpha$	$\epsilon^{-1}\alpha^{-0.5}R^p d^{1.5} \log^{1.5} n$	$\alpha^{-1} d \log^2 n$	nd	nd
Theorem F.2	1	$\epsilon^{-1}K_p(n)R^p d^{1.5}$	$d(p + \log n)$	$p(n + n^{1/p})d$	$pn^{1/p}d$

Table 3: Bounds for ℓ_p^p queries with $X \subset [0, R]^d$ and fixed $y \in [0, R]^d$. Errors are in expectation and privacy is pure ϵ-DP. The first row gives the general-power construction of [9] for fixed integer p, with a constant-factor accuracy-parameter rescaling, suppressing O_p constants. The second suppresses only absolute big-O constants and applies to every positive integer p. Here O_p permits dependence on p, and $\alpha \in (0, 1)$. Theorem F.2 also gives a query-dependent bound for arbitrary $y \in \mathbb{R}^d$.

Remark F.1. For each fixed integer p, $K_p(n) = O_p(\log^{3/2} n)$, so our bounded-query guarantee has no multiplicative slack or $\alpha^{-1/2}$ additive-error factor. The query time is $O_p(d \log n)$ and initialization is $O_p(nd)$. These comparisons concern the general-power construction in Table 3; [9] also gives a separate method for $p = 2$. We do not claim a uniform improvement as p grows: the explicit factor satisfies $K_p(n) \leq (p + 1)h\sqrt{m}\, 2^p$.

We first extend our key observation Lemma 3.1 with binomial expansion.

Lemma F.1. For an integer $p \geq 1$, a collection of numbers $\{x_1, x_2, \cdots, x_n\} \subset \mathbb{R}$, and $y \in \mathbb{R}$, we define two sets

$$S_+ := \{k \in [n] \ : \ x_k > y\}$$
$$S_- := \{k \in [n] \ : \ x_k < y\},$$

It holds

$$\sum_{k=1}^{n} |x_k - y|^p = \sum_{j=0}^{p} \binom{p}{j} y^{p-j}((-1)^{p-j} \sum_{k \in S_+} x_k^j + (-1)^j \sum_{k \in S_-} x_k^j),$$

where $\binom{p}{j}$ denotes the binomial coefficient that $\binom{p}{j} = \frac{p!}{j!(p-j)!}$.

Proof. We show that

$$\begin{aligned}
\sum_{k=1}^{n} |x_k - y|^p &= \sum_{k \in S_+} (x_k - y)^p + \sum_{k \in S_-} (y - x_k)^p \\
&= (\sum_{k \in S_+} \sum_{j=0}^{p} (-1)^{p-j} \binom{p}{j} x_k^j y^{p-j}) + (\sum_{k \in S_-} \sum_{j=0}^{p} (-1)^j \binom{p}{j} x_k^j y^{p-j}) \\
&= \sum_{j=0}^{p} (\binom{p}{j} (-1)^{p-j} y^{p-j} \sum_{k \in S_+} x_k^j) + \sum_{j=0}^{p} (\binom{p}{j} (-1)^j y^{p-j} \sum_{k \in S_-} x_k^j)
\end{aligned}$$

$$= \sum_{j=0}^{p} \binom{p}{j} y^{p-j}((-1)^{p-j} \sum_{k \in S_+} x_k^j + (-1)^j \sum_{k \in S_-} x_k^j).$$

Thus, we complete the proof. □

Lemma F.1 presents a decomposition of the final answer. To calculate the final answer, each node stores $\sum_{x_k \in I} x_k^q$ for $q = 0, \ldots, p$, with $x_k^0 := 1$, including when $x_k = 0$. The released left and right aggregates approximate $\sum_{k \in S_-} x_k^q$ and $\sum_{k \in S_+} x_k^q$, excluding the leaf containing an interior query. Queries outside the interval use the released root moments and public count n.

For the utility bound, let $Q(y) := \{0, \ldots, p\}$ and $r(y) := L$ when $0 < y < R$; otherwise let $Q(y) := \{1, \ldots, p\}$ and $r(y) := 1$. Define

$$V_p(y) := r(y) \sum_{q \in Q(y)} \binom{p}{q}^2 |y|^{2(p-q)} R^{2q},$$

with zeroth powers equal to one. The exterior and endpoint cases omit $q = 0$ because the root formula uses the public count. We count real arithmetic operations and random samples at unit cost.

Theorem F.1 (1-Dimensional ℓ_p^p DP Distance Query)**.** Given $X \subset [0, R]$ with public $n = |X| \geq 1$, $R > 0$, and integer $p \geq 1$, Algorithm 4 uses $O(pn^{1/p})$ space and supports the following operations:

- **INIT**(X, ϵ)**.** For $\epsilon \in (0, 1)$, it takes $O(p(n + n^{1/p}))$ time to build the data structure.
- **QUERY**$(y \in \mathbb{R})$**.** For each fixed y independent of the release randomness, it takes $O(p + \log n)$ time to output A. With $A' := \sum_{x \in X} |x - y|^p$,

$$\mathbb{E}[|A - A'|] \leq R^p + \frac{2\sqrt{2}(p+1)h}{\epsilon} \sqrt{V_p(y)}.$$

The data structure is ϵ-DP under replace-one adjacency. For $y \in [0, R]$, the error is at most $R^p + 2\sqrt{2}K_p(n)R^p/\epsilon$.

Proof. We set $L = \lceil (\log_2 n)/p \rceil$ and $h = L + 1$.

Init Time. Since $n^{1/p} \leq 2^L < 2n^{1/p}$, the tree has $O(n^{1/p})$ nodes, each storing $p + 1$ moments. For each record, we compute $1, x_k, \ldots, x_k^p$ by successive multiplication and add all $p + 1$ values to its leaf, taking $O(np)$ time in total. Combining child moments and adding noise takes $O(pn^{1/p})$ time. Thus initialization takes $O(p(n + n^{1/p}))$ time and storage takes $O(pn^{1/p})$ space.

Query Time. Each interior query visits L layers and updates $p + 1$ aggregates at each layer. Endpoint and exterior queries use the root moments. Computing powers of y successively and binomial coefficients through $\binom{p}{q+1} = \binom{p}{q}(p - q)/(q + 1)$ takes $O(p)$ operations. The total is $O(p(L + 1)) = O(p + \log n)$, including when $L = 0$ or $p > \log_2 n$.

Privacy Guarantees. For $q = 0, \ldots, p$, let $F_q : [0, R]^n \to \mathbb{R}^{2^{L+1}-1}$ be the vector of noiseless q-th moments. A replacement affects at most two paths of h nodes. Each removed or added contribution has magnitude at most R^q, so F_q has ℓ_1 sensitivity at most $2hR^q$. Independent coordinate-wise Laplace noise of scale $2(p + 1)hR^q/\epsilon$ makes the q-th release $\epsilon/(p + 1)$-DP. Lemma 2.1 gives total privacy budget $(p + 1)\epsilon/(p + 1) = \epsilon$. Queries are post-processing of these releases and public parameters.

Error Guarantees. Write $A - A' = N - B$. For $0 < y < R$, the omitted leaf contributes

$$\begin{aligned} B &= \sum_{x_k \in I_{L, j_L(y)}} |x_k - y|^p && \text{(Omitted leaf contribution)} \\ &\leq n \left(\frac{R}{2^L}\right)^p && \text{(Leaf width is } R/2^L\text{)} \\ &\leq R^p. && \text{(}pL \geq \log_2 n\text{)} \end{aligned}$$

Algorithm 4 ℓ_p^p Distance Query

data structure ℓ_p^p FASTERTREE ▷ Theorem F.1
members
  $T := \{s_{l,j,q}\}$ and public n, R, p, L, h ▷ Moment q of interval $I_{l,j}$
end members

procedure INIT(X, n, ϵ, R, p)
  $L \leftarrow \lceil (\log_2 n)/p \rceil$, $h \leftarrow L + 1$; initialize all $s_{l,j,q}$ to zero
  for each x_k in X **do**
    $j \leftarrow j_L(x_k) = \min\{2^L - 1, \lfloor 2^L x_k / R \rfloor\}$
    for q from 0 to p **do**
      $s_{L,j,q} \leftarrow s_{L,j,q} + x_k^q$
    end for
  end for
  for each layer l from $L-1$ down to 0 **do**
    for $j = 0, \ldots, 2^l - 1$ **do**
      for q from 0 to p **do**
        $s_{l,j,q} \leftarrow s_{l+1,2j,q} + s_{l+1,2j+1,q}$
      end for
    end for
  end for
  for q from 0 to p **do**
    Add independent noises drawn from $\mathsf{Laplace}(2(p+1)hR^q/\epsilon)$ to each $s_{l,j,q}$
  end for
  $T \leftarrow \{s\}$
end procedure

procedure QUERY($y \in \mathbb{R}$)
  for q from 0 to p **do**
    $s_{\text{left},q}, s_{\text{right},q} \leftarrow 0$
  end for
  if $y \le 0$ **then**
    $s_{\text{right},0} \leftarrow n$, $s_{\text{right},q} \leftarrow s_{0,0,q}$ for $q = 1, \ldots, p$
  else if $y \ge R$ **then**
    $s_{\text{left},0} \leftarrow n$, $s_{\text{left},q} \leftarrow s_{0,0,q}$ for $q = 1, \ldots, p$
  else
    for each layer l from 1 to L **do**
      $j \leftarrow j_l(y) = \min\{2^l - 1, \lfloor 2^l y / R \rfloor\}$
      for q from 0 to p **do**
        if $j \mod 2 = 1$ **then** ▷ if node j is the right child of its parent
          $s_{\text{left},q} \leftarrow s_{\text{left},q} + s_{l,j-1,q}$
        else ▷ if node j is the left child of its parent
          $s_{\text{right},q} \leftarrow s_{\text{right},q} + s_{l,j+1,q}$
        end if
      end for
    end for
  end if
  return $\sum_{q=0}^{p} \binom{p}{q} y^{p-q}((-1)^{p-q} s_{\text{right},q} + (-1)^q s_{\text{left},q})$ ▷ Lemma F.1
end procedure
end data structure

For $y \le 0$ or $y \ge R$, the root formula has $B = 0$.

Let $\lambda_q := 2(p+1)hR^q/\epsilon$. For $q \in Q(y)$, the noise N_q in the signed q-th aggregate is a sum of exactly $r(y)$ independent Laplace noises of scale λ_q. Thus $\mathbb{E}[N_q] = 0$ and $\mathrm{Var}[N_q] = 2r(y)\lambda_q^2$. In particular, Fact D.1 gives

$$\mathbb{E}[|N_q|] \le \sqrt{\mathrm{Var}[N_q]} \qquad \big(\mathbb{E}[N_q] = 0\big)$$

$$\leq \sqrt{2m}\lambda_q. \qquad \text{(Independent Laplace noises)}$$

Hence,

$$\mathbb{E}[|N_q|] \leq \frac{2\sqrt{2}(p+1)h\sqrt{m}R^q}{\epsilon}. \tag{F.1}$$

The moments receive independent noises, so their variances add even though their noiseless values depend on the same records. The binomial formula gives

$$\begin{aligned}\mathrm{Var}[N] &= 2r(y)\sum_{q\in Q(y)}\binom{p}{q}^2 |y|^{2(p-q)}\lambda_q^2 && \text{(Independent moment noises)}\\ &= \frac{8(p+1)^2h^2}{\epsilon^2}V_p(y). && \text{(Definitions of } \lambda_q \text{ and } V_p\text{)}\end{aligned}$$

Consequently, $\mathbb{E}[|A-A'|] \leq B + \sqrt{\mathrm{Var}[N]}$ gives the stated bound. For $y \in [0,R]$, each factor $|y|^{2(p-q)}R^{2q}$ is at most R^{2p}, and $\sum_{q=0}^{p}\binom{p}{q}^2 = \binom{2p}{p}$. Hence $V_p(y) \leq mR^{2p}\binom{2p}{p}$, which proves the bounded-query claim. When $n=1$, an interior query has $L=0$ and no noise contribution; its omitted leaf still obeys $B \leq R^p$. □

For high dimensional ℓ_p^p queries, we follow the coordinate-wise composition in Section 3.6, using Algorithm 4 in place of the one-dimensional ℓ_1 primitive. We build one independent moment tree per coordinate with budget ϵ/d and sum their query outputs.

Theorem F.2 (d-Dimensional ℓ_p^p DP Distance Query)**.** Given $X \subset [0,R]^d$ with public $n = |X| \geq 1$, $R > 0$, and integer $p \geq 1$, the coordinate-wise construction uses $O(pn^{1/p}d)$ space and supports the following operations:

- **INIT**(X,ϵ)**.** For $\epsilon \in (0,1)$, it takes $O(p(n+n^{1/p})d)$ time.
- **QUERY**$(y \in \mathbb{R}^d)$**.** For each fixed y independent of the release randomness, it takes $O(d(p+\log n))$ time to output A. With $A' := \sum_{x\in X}\|x-y\|_p^p$,

$$\mathbb{E}[|A-A'|] \leq dR^p + \frac{2\sqrt{2}(p+1)hd}{\epsilon}\sqrt{\sum_{i=1}^{d}V_p(y_i)}.$$

The data structure is ϵ-DP under replace-one adjacency. For $y \in [0,R]^d$, the error is at most $dR^p + 2\sqrt{2}K_p(n)R^p d^{3/2}/\epsilon$.

Proof. The time and space bounds follow by applying Theorem F.1 in each coordinate. A replacement changes at most one record in each coordinate dataset, so basic composition gives total privacy budget $d(\epsilon/d) = \epsilon$.

Write the coordinate errors as $N_i - B_i$. The one-dimensional proof, with budget ϵ/d, gives $0 \leq B_i \leq R^p$, $\mathbb{E}[N_i] = 0$, and $\mathrm{Var}[N_i] = 8(p+1)^2h^2d^2V_p(y_i)/\epsilon^2$. Independence across coordinates therefore yields

$$\begin{aligned}\mathbb{E}[|A-A'|] &\leq \sum_{i=1}^{d}B_i + \sqrt{\sum_{i=1}^{d}\mathrm{Var}[N_i]} && \text{(Triangle inequality and Fact D.1)}\\ &\leq dR^p + \frac{2\sqrt{2}(p+1)hd}{\epsilon}\sqrt{\sum_{i=1}^{d}V_p(y_i)}. && \text{(Coordinate bias and variance bounds)}\end{aligned}$$

For $y \in [0,R]^d$, substitute $V_p(y_i) \leq mR^{2p}\binom{2p}{p}$. This also gives the absolute-constant big-O bound in Table 3, since $d \geq 1$ and $\epsilon < 1$. □

G Illustrations of Sum and Prefix Aggregation

Both our algorithm and that of Backurs et al. [9] use a balanced binary-tree topology, as discussed in Sections 2.2 and 2.3. Algorithm 1 stores an interval count and coordinate sum at each node; Algorithm 2 combines these statistics using Lemma 3.1. The change is in the node statistics and query procedure, not a new tree topology. Figures 5–6 illustrate generic sum and exclusive-prefix representations for the values [1, 2, 3, 4], rather than the released statistics of the two DP algorithms.

- **The sum tree** is shown in Figure 5. Each leaf stores the original value, and each intermediate node stores the sum of its two children.
- **The exclusive-prefix tree** is shown in Figure 6. Each intermediate node stores the prefix-sum immediately to the left of all its descendant leaves. For example, node c stores $1+2=3$. Each leaf stores the sum of original values strictly to its left; for example, leaf g stores $1+2+3=6$.

For details on prefix-sum trees, we refer readers to the following standard references. The first is by CMU professor Guy Blelloch's lecture notes, "Prefix Sums and Their Applications" [91]. The second is the textbook "Programming Massively Parallel Processors: A Hands-On Approach" [92]. The last is the classic algorithms textbook, "Introduction to Algorithms" (3rd Ed.) [93].

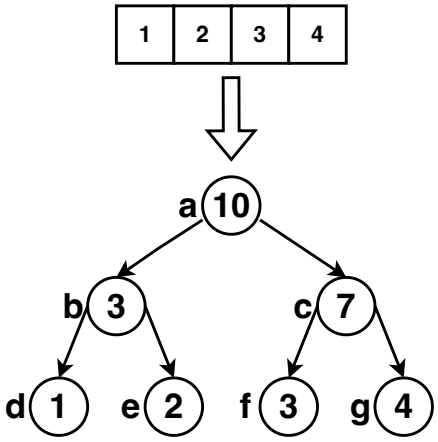


Figure 5: Generic sum tree

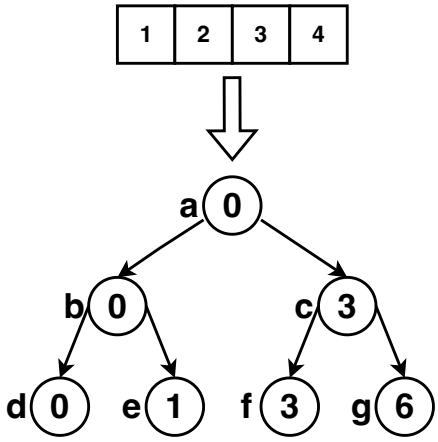


Figure 6: Generic exclusive-prefix tree

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We are very precise about our main results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please see Section C.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We list all assumptions and provide all proofs in step-by-step fashion.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please see Section A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We submitted our code with this draft.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Please see our Section A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Please see our plots in Section A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Yes, we specified compute resources in our Section A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We asked all authors to read through carefully.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please see our impact statement.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Given the formal nature of this work, we do not expect such concerns.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Please see our Section A.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: Given the formal nature of this work, we do not expect such concerns.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Given the formal nature of this work, we do not expect such concerns.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Given the formal nature of this work, we do not expect such concerns.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Please see our LLM usage statement at the beginning of the appendix.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.