Private Isotonic Regression

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this paper we consider the problem of differentially private (DP) algorithms for isotonic regression. For the most general problem of isotonic regression over a partially ordered set (poset) $\mathcal{X}$ and for any Lipschitz loss function, we obtain a pure-DP algorithm that, given n input points, has an expected excess empirical risk of roughly $\mathrm{width}(\mathcal{X}) \cdot \log |\mathcal{X}|/n$, where $\mathrm{width}(\mathcal{X})$ is the width of the poset. In contrast, we also obtain a near-matching lower bound of roughly $(\mathrm{width}(\mathcal{X}) + \log |\mathcal{X}|)/n$, that holds even for approximate-DP algorithms. Moreover, we show that the above bounds are essentially the best that can be obtained without utilizing any further structure of the poset. In the special case of a totally ordered set and for ℓ_1 and ℓ_2^2 losses, our algorithm can be implemented in near-linear running time; we also provide extensions of this algorithm to the problem of private isotonic regression with additional structural constraints on the output function.

1 Introduction

Isotonic regression is a basic primitive in statistics and machine learning, which has been studied at least since the 1950s [3, 8]; see also the textbooks on the topic [4, 37]. It has seen applications in numerous fields including medicine [30, 38] where the expression of an antigen is modeled as a monotone function of the DNA index and WBC count, and education [18], where isotonic regression was used to predict college GPA using high school GPA and standardized test scores. Isotonic regression is also arguably the most common non-parametric method for calibrating machine learning models [50], including modern neural networks [22].

In this paper, we study isotonic regression with a differential privacy (DP) constraint on the output model. DP [16, 15] is a highly popular notion of privacy for algorithms and machine learning primitives, and has seen increased adoption due to its powerful guarantees and properties [36, 42]. Despite the plethora of work on DP statistics and machine learning (see Section 5 for related work), ours is, to the best of our knowledge, the first to study DP isotonic regression.

In fact, we consider the most general version of the isotonic regression problem. We first set up some notation to describe our results. Let $(\mathcal{X}, \leq)$ be any partially ordered set (*poset*). A function $f : \mathcal{X} \to [0, 1]$ is *monotone* if and only if $f(x) \leq f(x')$ for all $x \leq x'$. For brevity, we use $\mathcal{F}(\mathcal{X}, \mathcal{Y})$ to denote the set of all monotone functions from $\mathcal{X}$ to $\mathcal{Y}$; throughout, we consider $\mathcal{Y} \subseteq [0, 1]$.

Let $[n]$ denote $\{1, \ldots, n\}$. Given an input dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times [0, 1])^n$, let the *empirical risk* of a function $f : \mathcal{X} \to [0, 1]$ be $\mathcal{L}(f; D) := \frac{1}{n} \sum_{i \in [n]} \ell(f(x_i), y_i)$, where $\ell : [0, 1] \times [0, 1] \to \mathbb{R}$ is a *loss function*.

We study private isotonic regression in the the basic machine learning framework of empirical risk minimization. Specifically, the goal of the *isotonic regression* problem, given dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times [0, 1])^n$, is to find a monotone function $f : \mathcal{X} \to [0, 1]$

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

that minimizes $\mathcal{L}(f; D)$. The *excess empirical risk* of a function f is defined as $\mathcal{L}(f; D) - \mathcal{L}(f^*; D)$ where $f^* := \operatorname{argmin}_{g \in \mathcal{F}(\mathcal{X}, \mathcal{Y})} \mathcal{L}(g; D)$.

1.1 Our Results

General Posets. Our first contribution is to give nearly tight upper and lower bounds for any poset, based on its *width*, as stated below (see Section 4 for the definition.)

Theorem 1 (Upper Bound for General Poset)**.** *Let $\mathcal{X}$ be any finite poset and let ℓ be an L-Lipschitz loss function. For any $\varepsilon \in (0, 1]$, there is an ε-DP algorithm for isotonic regression for ℓ with expected excess empirical risk at most $O\left(\frac{L \cdot \mathrm{width}(\mathcal{X}) \cdot \log|\mathcal{X}| \cdot (1+\log^2(\varepsilon n))}{\varepsilon n}\right)$.*[1]

Theorem 2 (Lower Bound for General Poset; Informal)**.** *For any $\varepsilon \in (0, 1]$ and any $\delta < 0.01 \cdot \varepsilon / |\mathcal{X}|$, any (ε, δ)-DP algorithm for isotonic regression for "nice" loss function ℓ must have expected excess empirical risk $\Omega\left(\frac{\mathrm{width}(\mathcal{X}) + \log|\mathcal{X}|}{\varepsilon n}\right)$.*

While our upper and lower bounds do not exactly match because of the multiplication-vs-addition of $\log|\mathcal{X}|$, we show in Section 4.3 that there are posets for which each bound in tight. In other words, this gap cannot be closed for generic posets.

Totally Ordered Sets. The above upper and lower bounds immediately translate to the case of totally ordered sets, by plugging in $\mathrm{width}(\mathcal{X}) = 1$. More importantly, we give efficient algorithms in this case, which runs in time $\tilde{O}(n^2 + n \log|\mathcal{X}|)$ for general loss function ℓ, and in nearly linear $\tilde{O}(n \cdot \log|\mathcal{X}|)$ time for the widely-studied ℓ_2^2 and ℓ_1 losses[2].

Theorem 3. *For all finite totally ordered sets $\mathcal{X}$, L-Lipschitz loss functions ℓ, and $\varepsilon \in (0, 1]$, there is an ε-DP algorithm for isotonic regression for ℓ with expected excess empirical risk $O\left(\frac{L \cdot (\log|\mathcal{X}|) \cdot (1+\log^2(\varepsilon n))}{\varepsilon n}\right)$. The running time of this algorithm is $\tilde{O}(n^2 + n \log|\mathcal{X}|)$ in general and can be improved to $\tilde{O}(n \log|\mathcal{X}|)$ for ℓ_1 and ℓ_2^2 losses.*

We are not aware of any prior work on private isotonic regression. A simple baseline algorithm for this problem would be to use the exponential mechanism over the set of all monotone functions taking values in a discretized set, to choose one with small loss. We show in the Supplementary Material that this achieves an excess empirical risk of $O(L \cdot \sqrt{\mathrm{width}(\mathcal{X}) \cdot \log|\mathcal{X}|/\varepsilon n})$, which is quadratically worse than the bound in Theorem 1. Moreover, even in the case of a totally ordered set, it is unclear how to implement such a mechanism efficiently.

We demonstrate the flexibility of our techniques by showing that it can be extended to variants of isotonic regression where, in addition to monotonicity, we also require f to satisfy additional properties. For example, we may want f to be L_f-Lipchitz for some specified $L_f > 0$. Other constraints we can handle include k-piecewise constant, k-piecewise linear, convexity, and concavity. For each of these constraints, we devise an algorithm that yields essentially the same error compared to the unconstrained case and still runs in polynomial time.

Theorem 4. *For all finite totally ordered sets $\mathcal{X}$, L-Lipschitz loss functions ℓ, and $\varepsilon \in (0, 1]$, there is an ε-DP algorithm for k-piecewise constant, k-piecewise linear, Lipchitz, convex, or concave isotonic regression for ℓ with expected excess empirical risk $\tilde{O}\left(\frac{L \cdot (\log|\mathcal{X}|)}{\varepsilon n}\right)$. The running time of this algorithm is $n^{O(1)} \cdot \log|\mathcal{X}|$.*

Organization. We next provide necessary background on DP. In Section 3, we prove our results for totally ordered sets (including Theorem 3). We then move on to discuss general posets in Section 4. Section 5 contains additional related work. Finally, we conclude with some discussion in Section 6. Due to space constraints, we omit some proofs from the main body; these can be found in the Supplementary Material.

[1] The excess risk bound is actually $O\left(\frac{L \cdot \mathrm{width}(\mathcal{X}) \cdot \log(|\mathcal{X}|/\mathrm{width}(\mathcal{X})) \cdot (1+\log^2(\varepsilon n))}{\varepsilon n}\right)$, which can be better that our stated bound when $\mathrm{width}(\mathcal{X}) \approx |\mathcal{X}|$. However, we skip this improved bound for simplicity.

[2] Recall that the ℓ_2^2 loss is $\ell_2^2(y, y') = (y - y')^2$ and the ℓ_1 loss is $\ell_1(y, y') = |y - y'|$.

2 Background on Differential Privacy

Two datasets $D, D' \subseteq \mathcal{X} \times \mathcal{Y}$ are said to be *neighboring*, denoted $D \sim D'$ if $D \subseteq D'$ with $|D' \setminus D| = 1$ or vice versa. We recall the formal definition of differential privacy [17, 15]:

Definition 5 (Differential Privacy (DP) [17, 15])**.** *Let $\varepsilon > 0$ and $\delta \in [0, 1]$. A randomized algorithm $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ is* (ε, δ)-differentially private *((ε, δ)-DP) if, for all $D \sim D'$ and all (measurable) outcomes $S \subseteq \mathcal{Y}$, we have that* $\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(D') \in S] + \delta$.

We denote $(\varepsilon, 0)$-DP as ε-DP (aka *pure*-DP). The case when $\delta > 0$ is referred to as *approximate*-DP.

Lemma 6. *(ε, δ)-DP satisfies the following:*

- **Basic Composition:** *If mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_t$ are such that $\mathcal{M}_i$ satisfies $(\varepsilon_i, \delta_i)$-DP, then the composed mechanism $(\mathcal{M}_1(D), \ldots, \mathcal{M}_t(D))$ satisfies $(\sum_i \varepsilon_i, \sum_i \delta_i)$-DP. This holds even under adaptive composition, where each $\mathcal{M}_i$ can depend on the outputs of $\mathcal{M}_1, \ldots, \mathcal{M}_{i-1}$.*
- **Parallel Composition [32]:** *If a mechanism $\mathcal{M}$ satisfies (ε, δ)-DP, then for any partition of $D = D_1 \sqcup \cdots \sqcup D_t$, the parallel composition given as $(\mathcal{M}(D_1), \ldots, \mathcal{M}(D_t))$ satisfies (ε, δ)-DP.*

Exponential Mechanism. The exponential mechanism solves the basic task of *selection* in data analysis: given a dataset $D \in \mathcal{Z}^n$ and a set $\mathcal{A}$ of options, it outputs the (approximately) best option, where "best" is defined by a *scoring function* $\mathfrak{s} : \mathcal{A} \times \mathcal{Z}^n \to \mathbb{R}$. The *$\varepsilon$-DP exponential mechanism* [33] is the randomized mechanism $\mathcal{M} : \mathcal{Z}^n \to \mathcal{A}$ given by

$$\forall\, D \in \mathcal{Z}^n,\ a \in \mathcal{A} \ :\ \Pr[\mathcal{M}(D) = a] \ \propto\ \exp\left(-\tfrac{\varepsilon}{2\Delta_{\mathfrak{s}}} \cdot \mathfrak{s}(a, D)\right),$$

where $\Delta_{\mathfrak{s}} := \sup_{D \sim D'} \max_{a \in \mathcal{A}} |\mathfrak{s}(a, D) - \mathfrak{s}(a, D')|$ is the *sensitivity* of the scoring function.

Lemma 7 ([33])**.** *For $\mathcal{M}$ being the ε-DP exponential mechanism, it holds for all $D \in \mathcal{Z}^n$ that*

$$\mathbb{E}[\mathfrak{s}(\mathcal{M}(D), D)] \ \leq\ \min_{a \in \mathcal{A}} \mathfrak{s}(a, D) \ +\ \tfrac{2\Delta_{\mathfrak{s}}}{\varepsilon} \log |\mathcal{A}|.$$

Lower Bound for Privatizing Vectors. Lower bounds for DP algorithms that can output a binary vector that is close (say, in the Hamming distance) to the input are well-known.

Lemma 8 (e.g., [31])**.** *Let $\varepsilon, \delta > 0$ and $m \in \mathbb{N}$. Then, for any (ε, δ)-DP algorithm $\mathcal{M} : \{0, 1\}^m \to \{0, 1\}^m$, we have $\mathbb{E}_{\mathbf{z} \sim \{0,1\}^m}[\|\mathcal{M}(\mathbf{z}) - \mathbf{z}\|_0] \geq e^{-\varepsilon} \cdot m \cdot 0.5 \cdot (1 - \delta)$.*

It is also simple to extend the lower bound for the case where the vector is not binary, as stated below. We defer the full proof to Appendix D.

Lemma 9. *Let D, m be any positive integer such that $D \geq 2$, let the input domain be $[D]^m$ and let two vectors $\mathbf{z}, \mathbf{z}' \in [D]^m$ be neighbors if and only if $\|\mathbf{z} - \mathbf{z}'\|_0 \leq 1$. Then, for any $(\ln(D/2), 0.25)$-DP algorithm $\mathcal{M} : [D]^m \to [D]^m$, we have that $\mathbb{E}_{\mathbf{z} \sim [D]^m}[\|\mathcal{M}(\mathbf{z}) - \mathbf{z}\|_0] \geq \Omega(m)$.*

Group Differential Privacy. For any neighboring relation $\sim$, we write $\sim_k$ as a neighboring relation where $D \sim_k D'$ iff there is a sequence $D = D_0, \ldots, D_{k'} = D'$ for some $k' \leq k$ such that $D_{i-1} \sim D_i$ for all $i \in [k']$.

Fact 10 (e.g., [40])**.** *Let $\varepsilon, \delta > 0$ and $k \in \mathbb{N}$. Suppose that $\mathcal{M}$ is an (ε, δ)-DP algorithm for the neighboring relation $\sim$. Then $\mathcal{M}$ is $\left(k\varepsilon, \frac{e^{k\varepsilon} - 1}{e^{\varepsilon} - 1} \cdot \delta\right)$-DP for the neighboring relation $\sim_k$.*

3 DP Isotonic Regression over Total Orders

We first focus on the "one-dimensional" case where $\mathcal{X}$ is totally ordered; for convenience, we assume that $\mathcal{X} = [m]$ where the order is the natural order on integers. We first present an efficient algorithm for the this case and then a matching lower bound.

3.1 An Efficient Algorithm

To describe our algorithm, it will be more convenient to use the unnormalized version of the empirical risk, which we define as $\mathcal{L}^{\text{abs}}(f; D) := \sum_{(x,y) \in D} \ell(f(x), y)$.

Algorithm 1 DP Isotonic Regression for Totally Ordered Sets.

Input: $\mathcal{X} = [m]$, dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, DP parameter ε.
Output: Monotone function $f : [m] \to [0, 1]$.

$T \leftarrow \lceil \log(\varepsilon n) \rceil$
$\varepsilon' \leftarrow \varepsilon / T$
$P_{0,0} \leftarrow [m]$

for $t = 0, \ldots, T-1$ **do**
  for $i = 0, \ldots, 2^t - 1$ **do**
    ▷ $D_{i,t} \leftarrow \{(x_j, y_j) \mid j \in [n], x_j \in P_{i,t}\}$ {Set of all input points whose x belongs to $P_{i,t}$}
    {Notation: Define $D_{i,t}^{\leq \alpha} := \{(x, y) \in D_{i,t} \mid x \leq \alpha\}$ and $D_{i,t}^{>\alpha}$ similarly }
    ▷ Choose threshold $\alpha_{i,t} \in P_{i,t}$, using ε'-DP exponential mechanism with scoring function

$$\begin{aligned}\mathrm{score}_{i,t}(\alpha) \;:=\; & \min_{f_1 \in \mathcal{F}(P_{i,t}^{\leq \alpha}, [\frac{i}{2^t}, \frac{i+0.5}{2^t}])} \mathcal{L}^{\mathrm{abs}}_{[\frac{i}{2^t}, \frac{i+1}{2^t}]}(f_1; D_{i,t}^{\leq \alpha}) \\ & + \min_{f_2 \in \mathcal{F}(P_{i,t}^{>\alpha}, [\frac{(i+0.5)}{2^t}, \frac{(i+1)}{2^t}])} \mathcal{L}^{\mathrm{abs}}_{[\frac{i}{2^t}, \frac{i+1}{2^t}]}(f_2; D_{i,t}^{>\alpha})\end{aligned}$$

    {Note: $\mathrm{score}_{i,t}(\alpha)$ has sensitivity at most $L/2^t$. }
    ▷ $P_{2i,t+1} \leftarrow P_{i,t}^{\leq \alpha_{i,t}}$ and $P_{2i+1,t+1} \leftarrow P_{i,t}^{>\alpha_{i,t}}$.

Let $f : [m] \to [0, 1]$ be given as $f(x) = i/2^{T-1}$ for all $x \in P_{i,T-1}$ and all $i \in [2^t]$.
return f

We now provide a high-level overview of our algorithm. Any monotone function $f : [m] \to [0, 1]$ corresponds to a (not necessarily unique) threshold $\alpha \in [m]$ such that $f(a) \geq 1/2$ for all $a > \alpha$ and $f(a) \leq 1/2$ for all $a \leq \alpha$. Our algorithm works by first choosing this threshold α in a private manner using the exponential mechanism. The choice of α partitions $[m]$ into $[m]^{>\alpha} := \{a \in [m] \mid a > \alpha\}$ and $[m]^{\leq \alpha} := \{a \in [m] \mid a \leq \alpha\}$. The algorithm recurses on these two parts to find functions $f_> : [m]^{>\alpha} \to [1/2, 1]$ and $f_\leq : [m]^{\leq \alpha} \to [0, 1/2]$, which are glued to obtain the final function.

In particular, the algorithm proceeds in T stages, where in stage t, the algorithm starts with a partition of $[m]$ into 2^t intervals $\{P_{i,t} \mid i \in \{0, \ldots, 2^t - 1\}\}$, and the algorithm eventually outputs a monotone function f such that $f(x) \in [i/2^t, (i+1)/2^t]$ for all $x \in P_{i,t}$. This partition is further refined for stage $t+1$ by choosing a threshold $\alpha_{i,t}$ in $P_{i,t}$ and partitioning $P_{i,t}$ into $P_{i,t}^{>\alpha_{i,t}}$ and $P_{i,t}^{\leq \alpha_{i,t}}$. In the final stage, the function f is chosen to be the constant $i/2^{T-1}$ over $P_{i,T-1}$. Note that the algorithm may stop at $T = \Theta_\varepsilon(\log n)$ because the Lipschitzness of ℓ ensures that assigning each partition to the constant $i/2^{T-1}$ cannot increase the error by more than $L/2^T \leq O_\varepsilon(L/n)$.

We already have mentioned above that each $\alpha_{i,t}$ has to be chosen in a private manner. However, if we let the scoring function directly be the unnormalized empirical risk, then its sensitivity remains as large as L even at a large stage t. This is undesirable because the error from each run of the exponential mechanism can be as large as $O(L \cdot \log m)$ but there are as many as 2^t runs in stage t. Adding these error terms up would result in a far larger total error than desired.

To circumvent this, we observe that while the sensitivity can still be large, they are mostly "ineffective" because the function range is now restricted to only an interval of length $1/2^t$. Indeed, we may use the following "clipped" version of the loss function which has low sensitivity of $L/2^t$ instead.

Definition 11 (Clipped Loss Function)**.** *For a range $[\tau, \theta] \subseteq [0, 1]$, let $\ell_{[\tau,\theta]} : [\tau, \theta] \times [0, 1] \to \mathbb{R}$ be given as $\ell_{[\tau,\theta]}(\hat{y}, y) := \ell(\hat{y}, y) - \min_{y' \in [\tau,\theta]} \ell(y', y)$. Similar to above, we also define $\mathcal{L}^{\mathrm{abs}}_{[\tau,\theta]}(f; D) := \sum_{(x,y) \in D} \ell_{[\tau,\theta]}(f(x_i), y_i)$.*

Observe that $\mathrm{range}(\ell_{[\tau,\theta]}) \subseteq [0, L \cdot (\theta - \tau)]$, since ℓ is L-Lipschitz. In other words, the sensitivity of $\mathcal{L}^{\mathrm{abs}}_{[\tau,\theta]}(f; D)$ is only $L \cdot (\theta - \tau)$. The full description of our algorithm is presented in Algorithm 1.

Proof of Theorem 3. Before proceeding to prove the algorithm's privacy and utility guarantees, we note that the output f is indeed monotone since for every $x' < x$ that gets separated when we partition $P_{i,t}$ into $P_{2i,t+1}, P_{2i+1,t+1}$, we must have $x' \in P_{2i,t+1}$ and $x \in P_{2i+1,t+1}$.

Privacy Analysis. Since the exponential mechanism is ε'-DP and the dataset is partitioned with the exponential mechanism being applied only to each partition once, the parallel composition property (Lemma 6) implies that the entire subroutine for each t is ε'-DP. Thus, by basic composition (Lemma 6), it follows that Algorithm 1 is ε-DP (since $\varepsilon = \varepsilon' T$).

Utility Analysis. Since the sensitivity of $\mathrm{score}_{i,t}(\cdot)$ is at most $L/2^t$, we have from Lemma 7, that for all $t \in \{0, \ldots, T-1\}$ and $i \in \{0, 1, \ldots, 2^t\}$,

$$\mathbb{E}\left[\mathrm{score}_{i,t}(\alpha_{i,t}) - \min_{\alpha \in P_{i,t}} \mathrm{score}_{i,t}(\alpha)\right] \;\le\; O\left(\frac{L \cdot \log |P_{i,t}|}{\varepsilon' \cdot 2^t}\right) \;\le\; O\left(\frac{L \cdot \log m}{\varepsilon' \cdot 2^t}\right). \tag{1}$$

Let $h_{i,t}$ denote $\mathrm{argmin}_{h \in \mathcal{F}(P_{i,t}, [i/2^t, (i+1)/2^t])} \mathcal{L}^{\mathrm{abs}}(h; D_{i,t})$ (with ties broken arbitrarily). Then, let $\tilde{\alpha}_{i,t}$ denote the largest element in $P_{i,t}$ such that $h_{i,t}(\tilde{\alpha}_{i,t}) \le (i+1/2)/2^t$; namely, $\tilde{\alpha}_{i,t}$ is the optimal threshold when restricted to $D_{i,t}$. Under this notation, we have that

$$\begin{aligned}
&\mathrm{score}_{i,t}(\alpha_{i,t}) - \min_{\alpha \in P_{i,t}} \mathrm{score}_{i,t}(\alpha) \\
&\le\; \mathrm{score}_{i,t}(\alpha_{i,t}) - \mathrm{score}_{i,t}(\tilde{\alpha}_{i,t}) \\
&=\; \left(\mathcal{L}^{\mathrm{abs}}_{[i/2^t,(i+0.5)/2^t]}(h_{2i,t+1}; D_{2i,t+1}) + \mathcal{L}^{\mathrm{abs}}_{[(i+0.5)/2^t,(i+1)/2^t]}(h_{2i+1,t+1}; D_{2i+1,t+1})\right) \\
&\qquad -\; \mathcal{L}^{\mathrm{abs}}_{[i/2^t,(i+1)/2^t]}(h_{i,t}; D_{i,t}) \\
&=\; \mathcal{L}^{\mathrm{abs}}(h_{2i,t+1}; D_{2i,t+1}) + \mathcal{L}^{\mathrm{abs}}(h_{2i+1,t+1}; D_{2i+1,t+1}) - \mathcal{L}^{\mathrm{abs}}(h_{i,t}; D_{i,t}).
\end{aligned} \tag{2}$$

Finally, notice that

$$\mathcal{L}^{\mathrm{abs}}(f; D_{i,T-1}) - \mathcal{L}^{\mathrm{abs}}(h_{i,T-1}; D_{i,T-1}) \le \tfrac{L}{2^{T-1}} \cdot |D_{i,T-1}| = O\left(\tfrac{L \cdot |D_{i,T-1}|}{\varepsilon n}\right). \tag{3}$$

With all the ingredients ready, we may now bound the expected (unnormalized) excess risk:

$$\begin{aligned}
\mathcal{L}^{\mathrm{abs}}(f; D) \;&=\; \textstyle\sum_{0 \le i < 2^{T-1}} \mathcal{L}^{\mathrm{abs}}(f; D_{i,T-1}) \\
&\overset{(3)}{\le}\; \textstyle\sum_{0 \le i < 2^{T-1}} \left(O\left(\frac{L \cdot |D_{i,T-1}|}{\varepsilon n}\right) + \mathcal{L}^{\mathrm{abs}}(h_{i,T-1}; D_{i,T-1})\right) \\
&=\; O(L/\varepsilon) \;+\; \textstyle\sum_{0 \le i < 2^{T-1}} \mathcal{L}^{\mathrm{abs}}(h_{i,T-1}; D_{i,T-1}) \\
&=\; O(L/\varepsilon) \;+\; \mathcal{L}^{\mathrm{abs}}(h_{0,0}; D_{0,0}) \\
&\quad + \textstyle\sum_{\substack{t \in [T-1] \\ 0 \le i < 2^{t-1}}} \left(\mathcal{L}^{\mathrm{abs}}(h_{2i,t}; D_{2i,t}) + \mathcal{L}^{\mathrm{abs}}(h_{2i+1,t}; D_{2i+1,t}) - \mathcal{L}^{\mathrm{abs}}(h_{i,t-1}; D_{i,t-1})\right).
\end{aligned}$$

Taking the expectation on both sides and using (1) and (2) yields

$$\begin{aligned}
\mathbb{E}[\mathcal{L}^{\mathrm{abs}}(f; D)] \;&\le\; O(L/\varepsilon) \;+\; \mathcal{L}^{\mathrm{abs}}(h_{0,0}; D_{0,0}) \;+\; \textstyle\sum_{\substack{t \in [T-1] \\ 0 \le i < 2^{t-1}}} O\left(\frac{L \cdot \log m}{\varepsilon' \cdot 2^t}\right) \\
&=\; O(L/\varepsilon) \;+\; \mathcal{L}^{\mathrm{abs}}(f^*; D) \;+\; O\left(T^2 \cdot \frac{L \cdot \log m}{\varepsilon}\right) \\
&=\; \mathcal{L}^{\mathrm{abs}}(f^*; D) \;+\; O\left(\frac{L \cdot \log m \cdot (1 + \log^2(\varepsilon n))}{\varepsilon}\right).
\end{aligned}$$

Dividing both sides by n yields the desired claim.

Running Time. To obtain a bound on the running time for general loss functions, we need to make a slight modification to the algorithm: we will additionally only restrict the range of f_1, f_2 to multiples of $1/2^{T-1}$. We remark that this does not affect the utility since anyway we always take the final output whose values are multiples of $1/2^{T-1}$.

Given any dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_1 < \cdots < x_n$. The *prefix isotonic regression* algorithm is to compute, for each $i \in [n]$, the optimal loss in isotonic regression on

$(x_1, y_1), \ldots, (x_i, y_i)$. Straightforward dynamic programming solves this in $O(n \cdot v)$ time, where v denote the number of possible values allowed in the function.

Now, for each i, t, we may run the above algorithm with $D = D_{i,t}$ and the allowed values are all multiples of $1/2^{T-1}$ in $[\frac{i}{2^t}, \frac{i+0.5}{2^t}]$; this gives us $\min_{f_1 \in \mathcal{F}(P_{i,t}^{\leq \alpha}, [\frac{i}{2^t}, \frac{i+0.5}{2^t}])} \mathcal{L}^{\text{abs}}_{[\frac{i}{2^t}, \frac{i+1}{2^t}]}(f_1; D_{i,t}^{\leq \alpha})$ for all $\alpha \in P_{i,t}$ in time $O(|D_{i,t}| \cdot 2^{T-t} + |P_{i,t}|)$. Analogously, we can also compute $\min_{f_2 \in \mathcal{F}(P_{i,t}^{> \alpha}, [\frac{(i+0.5)}{2^t}, \frac{(i+1)}{2^t}])} \mathcal{L}^{\text{abs}}_{[\frac{i}{2^t}, \frac{i+1}{2^t}]}(f_2; D_{i,t}^{> \alpha})$ for all $\alpha \in P_{i,t}$ in a similar time. Thus, we can compute $(\text{score}_{i,t}(\alpha))_{\alpha \in P_{i,t}}$ in time $O(|D_{i,t}| \cdot 2^{T-t} + |P_{i,t}|)$, and then sample accordingly.

We can further speed up the algorithm by observing that the score remains constant for all $\alpha \in [x_i, x_{i+1})$. Hence, we may first sample an interval among $[0, x_1), [x_1, x_2), \ldots, [x_{n-1}, x_n), [x_n, m)$ and then sample $\alpha_{i,t}$ uniformly from that interval. This entire process can be done in $O(|D_{i,t}| \cdot 2^{T-t} + \log m)$ time. In total, the running time of the algorithm is thus

$$\sum_{t=0}^{T-1} \sum_{i=0}^{2^t-1} O(|D_{i,t}| \cdot 2^{T-t} + \log m) \leq \sum_{t=0}^{T-1} O(n 2^T + 2^t \cdot \log m) \leq O(n^2 \log n + n \log m).$$

Near-Linear Time Algorithms for ℓ_1, ℓ_2^2 Losses. We now describe faster algorithms for the ℓ_1 and ℓ_2^2 loss functions, thereby proving the last part of Theorem 3. The key observation is that for convex loss functions, the restricted optimal is simple: we just have to "clip" the optimal function to be in the range $[\tau, \theta]$. Below $\text{clip}_{[\tau,\theta]}$ denotes the function $y \mapsto \min\{\theta, \max\{\tau, y\}\}$.

Observation 12. *Let ℓ be any convex loss function, D any dataset, $f^* \in \text{argmin}_{f \in \mathcal{F}(\mathcal{X}, \mathcal{Y})} \mathcal{L}(f; D)$ and $\tau \leq \theta$ any real numbers such that $\tau, \theta \in \mathcal{Y}$. Define $f^*_{\text{clipped}}(x) := \text{clip}_{[\tau,\theta]}(f^*(x))$. Then, we must have $f^*_{\text{clipped}}(x) \in \text{argmin}_{f \in \mathcal{F}(\mathcal{X}, \mathcal{Y} \cap [\tau,\theta])} \mathcal{L}(f; D)$.*

We will now show how to compute all scores in Algorithm 1 simultaneously for all α (for fixed i, t) in nearly linear time. To do this, recall the prefix isotonic regression problem from earlier. For this problem, Stout [41] gave an $O(n)$-time algorithm for ℓ_2 loss and an $O(n \log n)$-time algorithm for ℓ_1 loss (both the unrestricted value case). Furthermore, after the ith iteration, the algorithm also keeps a succinct representation S_i^{opt} of the optimal solution in the form of an array $(i_1, v_1, \ell_1), \ldots, (i_k, v_k, \ell_k)$, which denotes $f(x) = v_j$ for all $x \in [x_{i_j}, x_{i_{j+1}})$, and ℓ_j indicates the loss $\mathcal{L}^{\text{abs}}$ up until $x_{i_{j+1}}$, not including.

We can extend the above algorithm to *prefix clipped isotonic regression* problem, which we define in the same manner as above except that we restrict the function range to be $[\tau, \theta]$ for some given $\tau < \theta$. Using Observation 12, it is not hard to extend the above algorithm to work in this case too.

Lemma 13. *There is an $O(n \log n)$-time algorithm for ℓ_2^2 and ℓ_1* prefix clipped isotonic regression.

Proof. We first precompute $c_\tau(i) = \sum_{j \leq i} \ell(\tau, x_j)$ and $c_\theta(i) = \sum_{j \leq i} \ell(\theta, x_j)$ for all $i \in [n]$. Then, run the aforementioned algorithm from [41]. At each iteration i, use binary search to find the largest index j_τ such that $v_{j_\tau} < \tau$ and the largest index j_θ such that $v_{j_\theta} < \theta$. Observation 12 implies that the optimal solution of the clipped version is simply the same as the unrestricted version except that we need to change the function values before x_{j_τ} to τ and after x_{j_θ} to θ. The loss of this clipped optimal can be written as $\ell_{j_\theta} - \ell_{j_\tau} + c_\tau(j_\tau) + (c_\theta(i) - c_\theta(j_\theta))$, which can be computed in $O(1)$ time given that we have already precomputed c_τ, c_θ. The running time of the entire algorithm is thus the same as that of [41] together with the binary search time; the latter totals to $O(n \log n)$. $\square$

Our fast algorithm for computing $(\text{score}_{i,t}(\alpha))_{\alpha \in P_{i,t}}$ first runs the above algorithm with $\tau = \frac{i}{2^t}, \theta = \frac{i+0.5}{2^t}$ and $D = D_{i,t}$; this gives us $\min_{f_1 \in \mathcal{F}(P_{i,t}^{\leq \alpha}, [\frac{i}{2^t}, \frac{i+0.5}{2^t}])} \mathcal{L}^{\text{abs}}_{[\frac{i}{2^t}, \frac{i+1}{2^t}]}(f_1; D_{i,t}^{\leq \alpha})$ for all $\alpha \in P_{i,t}$ in time $O(|D_{i,t}| \log |D_{i,t}| + |P_{i,t}|)$. Analogously, we can also compute $\min_{f_2 \in \mathcal{F}(P_{i,t}^{> \alpha}, [\frac{(i+0.5)}{2^t}, \frac{(i+1)}{2^t}])} \mathcal{L}^{\text{abs}}_{[\frac{i}{2^t}, \frac{i+1}{2^t}]}(f_2; D_{i,t}^{> \alpha})$ for all $\alpha \in P_{i,t}$ in a similar time. Thus, we can compute $(\text{score}_{i,t}(\alpha))_{\alpha \in P_{i,t}}$ in time $O(|D_{i,t}| \log |D_{i,t}| + |P_{i,t}|)$, and sample accordingly. Via the

same observation as the general loss function case, this can be sped up further to $O(|D_{i,t}| \log |D_{i,t}| + \log m)$ time. In total, the running time of the algorithm is thus

$$\sum_{t=0}^{T-1} \sum_{i=0}^{2^t-1} O(|D_{i,t}| \log |D_{i,t}| + \log m) \leq \sum_{t=0}^{T-1} O(n \log n + 2^t \log m) \leq O(n(\log^2 n + \log m)). \quad \square$$

3.2 A Nearly Matching Lower Bound

We show that the excess empirical risk guarantee in Theorem 3 is tight, even for approximate-DP algorithms with a sufficiently small δ, under a mild assumption about the loss function stated below.

Definition 14. *For $R \geq 0$, a loss function ℓ is said to be R-*distance-based *if there exist $g : [0,1] \to \mathbb{R}_+$ such that $\ell(y, y') = g(|y - y'|)$ where g is a non-decreasing function $g(0) = 0$ and $g(1/2) \geq R$.*

We remark that standard loss functions, including ℓ_1 or ℓ_2^2 loss, are all $\Omega(1)$-distance-based.

Our lower bound is stated below. It is proved via a packing argument [24] in a similar manner as a lower bound for properly PAC learning threshold functions [9]. This is not a coincidence: indeed, when we restrict the range of our function to $\{0, 1\}$, the problem becomes exactly (the empirical version of) properly learning threshold functions. As a result, the same technique can be used to prove a lower bound in our setting as well. The full details of this proof is in the SM.

Theorem 15. *For all $0 \leq \delta < 0.1 \cdot (e^\varepsilon - 1)/m$, any (ε, δ)-DP algorithm for isotonic regression over $[m]$ for any R-distance-based loss function ℓ must have expected excess empirical risk $\Omega\left(R \cdot \min\left\{1, \frac{\log m}{\varepsilon n}\right\}\right)$.*

3.3 Extensions

We now discuss several variants of the isotonic regression problem that places certain additional constraints on the function f that we seek, as listed below.

- k-Piecewise Constant: f must be a step function that consists of at most k pieces.
- k-Piecewise Linear: f must be a piecewise linear function with at most k pieces.
- Lipschitz Regression: f must be L_f-Lipschitz for some specified $L_f > 0$.
- Convex/Concave: f must be convex/concave.

We devise a general meta algorithm that, with a small tweak in each case, works for all of these constraints and yield Theorem 4. At a high-level, our algorithm is similar to Algorithm 1, except that, in addition to using exponential mechanism to pick the threshold $\alpha_{i,t}$, we also pick certain auxiliary information that is then passed onto the next stage. For example, in the k-piecewise constant setting, the algorithm in fact picks also the number of pieces to the left of $\alpha_{i,t}$ and that to the right of it. These are then passed on to the next stage. The algorithm stops when the number of pieces become one, and then simply use the exponential mechanism to find the constant value on this subdomain.

4 DP Isotonic Regression over General Posets

We now provide an algorithm and lower bounds for the case of general discrete posets. We first recall basic quantities about posets. An *anti-chain* of a poset $(\mathcal{X}, \leq)$ is a set of elements such that no two distinct elements are comparable, whereas a *chain* is a set of elements such that every pair of elements is comparable. The *width* of a poset $(\mathcal{X}, \leq)$, denoted by $\mathrm{width}(\mathcal{X})$, is defined as the maximum size among all anti-chains in the poset. The *height* of $(\mathcal{X}, \leq)$, denoted by $\mathrm{height}(\mathcal{X})$, is defined as the maximum size among all chains in the poset. Dilworth's theorem and Mirsky's theorem give the following relation between chains an anti-chains:

Lemma 16 (Dilworth's and Mirsky's theorems [11, 35])**.** *A poset with width w can be partitioned into w chains. A poset with height h can be partitioned into h anti-chains.*

4.1 An Algorithm

Our algorithm for general posets is similar to that of totally ordered set presented in the previous section. The only difference is that, instead of attempting to pick a single maximal point α

such that $f(\alpha) \leq \tau$ as in the previous case, there could now be many such maximal α's. Indeed, we need to use the exponential mechanism to pick all such α's. Since these are all maximal, they must be incomparable; therefore, they form an anti-chain. Since there can be as many as $|\mathcal{X}|^{\mathrm{width}(\mathcal{X})}$ anti-chains in total, this means that the error from the exponential mechanism is $O\left(\log |\mathcal{X}|^{\mathrm{width}(\mathcal{X})}/\varepsilon'\right) = O(\mathrm{width}(\mathcal{X}) \log |\mathcal{X}|/\varepsilon')$, leading to the multiplicative increase of $\mathrm{width}(\mathcal{X})$ in the total error. This completes our proof sketch for Theorem 1.

4.2 Lower Bounds

To prove a lower bound of $\Omega(\mathrm{width}(\mathcal{X})/\varepsilon n)$, we observe that the values of the function in any anti-chain can be arbitrary. Therefore, we may use each element in a maximum anti-chain to encode $\mathcal{X}$ as a binary vector. The lower bound from Lemma 8 then gives us $\Omega(\mathrm{width}(\mathcal{X})/n)$ lower bound for $\varepsilon = 1$, as formalized below.

Lemma 17. *For any $\delta > 0$, any $(1, \delta)$-DP algorithm for isotonic regression for any R-distance-based loss function ℓ must have expected excess empirical risk $\Omega\left(R(1-\delta) \cdot \min\left\{1, \frac{\mathrm{width}(\mathcal{X})}{n}\right\}\right)$.*

Proof. Consider any $(1, \delta)$-DP isotonic regression algorithm $\mathcal{M}'$ for loss ℓ. Let A be any maximum anti-chain (of size $\mathrm{width}(A)$) in $\mathcal{X}$. We use this algorithm to build a $(1, \delta)$-DP algorithm $\mathcal{M}$ for privatizing a binary vector of $m = \min\{n, |A| - 1\}$ dimensions as follows:

- Let $x_0, x_1, \ldots, x_m$ be distinct elements of A.
- On input $\mathbf{z} \in \{0,1\}^m$, create a dataset $D = \{(x_1, z_1), \ldots, (x_m, z_m), (x_0, 0), \ldots, (x_0, 0)\}$ where $(x_0, 0)$ is repeated $n - m$ times.
- Run $\mathcal{M}'$ on instance D to get f, and output a vector $\mathbf{z}'$ where $z'_i = \mathbf{1}[f(x_i) \geq 1/2]$.

It is obvious that this algorithm is $(1, \delta)$-DP. Observe also that $\mathcal{L}(f^*; D) = 0$ and thus $\mathcal{M}'$'s expected excess empirical risk is $\mathbb{E}_{f \sim \mathcal{M}'(D)}[\mathcal{L}(f; D)] \geq R \cdot \mathbb{E}_{\mathbf{z}' \sim \mathcal{M}(\mathbf{z})}[\|\mathbf{z} - \mathbf{z}'\|_0]/n$, which, from Lemma 8, must be at least $\Omega(Rm(1-\delta)/n) = \Omega\left(R(1-\delta) \cdot \min\left\{1, \frac{\mathrm{width}(\mathcal{X})}{n}\right\}\right)$. □

By using group privacy (Fact 10) and repeating each element $\Theta(1/\varepsilon)$ times, we arrive at a lower bound of $\Omega\left(R \cdot \min\left\{1, \frac{\mathrm{width}(\mathcal{X})}{\varepsilon n}\right\}\right)$. Furthermore, since $\mathcal{X}$ contains $\mathrm{height}(\mathcal{X})$ elements that form a totally ordered set, Theorem 15 gives a lower bound of $\Omega(R \cdot \log(\mathrm{height}(\mathcal{X}))/\varepsilon n)$ as long as $\delta < 0.01 \cdot \varepsilon/\mathrm{height}(\mathcal{X})$. Finally, due to Lemma 16, we have $\mathrm{height}(\mathcal{X}) \geq |\mathcal{X}|/\mathrm{width}(\mathcal{X})$, which means that $\max\{\mathrm{width}(\mathcal{X}), \log(\mathrm{height}(\mathcal{X})\} \geq \Omega(\log |\mathcal{X}|)$. Thus, we arrive at:

Theorem 18. *For any $\varepsilon \in (0, 1]$ and any $\delta < 0.01 \cdot \varepsilon/|\mathcal{X}|$, any (ε, δ)-DP algorithm for isotonic regression for R-distance-based loss function ℓ must have expected excess empirical risk $\Omega\left(R \cdot \min\left\{1, \frac{\mathrm{width}(\mathcal{X}) + \log |\mathcal{X}|}{\varepsilon n}\right\}\right)$.*

4.3 Tight Examples for Upper and Lower Bounds

Recall that our upper bound is $\tilde{O}\left(\frac{\mathrm{width}(\mathcal{X}) \cdot \log |\mathcal{X}|}{\varepsilon n}\right)$ while our lower bound is $\Omega\left(\frac{\mathrm{width}(\mathcal{X}) + \log |\mathcal{X}|}{\varepsilon n}\right)$. One might wonder whether this gap can be closed. Below we show that, unfortunately, this is impossible in general: there are posets for which each bound is tight.

Tight Lower Bound Example. Let $\mathcal{X}_{\mathrm{disj}(w,h)}$ denote the poset that consists of w disjoint chains, $C_1, \ldots, C_h$ where $|C_1| = h$ and $|C_2| = \cdots = |C_w| = 1$. (Every pair of elements on different chains are incomparable.) In this case, we can solve the isotonic regression problem directly on each chain and piece the solutions together into the final output f. Note that $|\mathcal{X}_{\mathrm{disj}(w,h)}| = (w-1)h + 1$ and $\mathrm{width}(\mathcal{X}) = w, \mathrm{height}(\mathcal{X}) = h$. According to Theorem 1, the unnormalized excess empirical risk in C_i is $\tilde{O}\left(\log(|C_i|)/\varepsilon\right)$. Therefore, the total (normalized) empirical risk for the entire domain $\mathcal{X}$ is $\tilde{O}\left(\frac{\log h + (w-1)}{\varepsilon n}\right)$. This is at most $\tilde{O}\left(\frac{w}{\varepsilon n}\right)$ as long as $h \leq \exp(O(w))$; this matches the lower bound.

Tight Upper Bound Example. Consider the grid poset $\mathcal{X}_{\mathrm{grid}(w,h)} := [w] \times [h]$ where $(x, y) \leq (x', y')$ if and only if $x \leq x'$ and $y \leq y'$. We assume throughout that $w \leq h$. Observe that $\mathrm{width}(\mathcal{X}_{\mathrm{grid}(w,h)}) = w$ and $\mathrm{height}(\mathcal{X}_{\mathrm{grid}(w,h)}) = w + h$.

We will show the following lower bound, which matches the $\tilde{O}\left(\frac{\mathrm{width}(\mathcal{X})\log|\mathcal{X}|}{\varepsilon n}\right)$ upper bound in the case where $h \geq w^{1+\Omega(1)}$, up to $O(\log^2(\varepsilon n)$ factor. We prove it by a reduction from Lemma 9. Note that this reduction is in some sense a "combination" of the proofs of Theorem 15 and Lemma 17, as the coordinate-wise encoding aspect of Lemma 17 is still present (across the rows) whereas the packing-style lower bound is present in how we embed elements of $[D]$ (in blocks of columns).

Lemma 19. *For any $\varepsilon \in (0,1]$ and $\delta < O_\varepsilon(1/h)$, any (ε,δ)-DP algorithm for isotonic regression for any R-distance-based loss function ℓ must have expected excess empirical risk $\Omega\left(R \cdot \min\left\{1, \frac{w\cdot\log(h/w)}{\varepsilon n}\right\}\right)$.*

Proof. Let $D := \lfloor h/w - 1\rfloor, m = w$ and $r := \min\{\lfloor 0.5n/m\rfloor, \lfloor 0.5\ln(D/2)/\varepsilon\rfloor\}$. Consider any (ε,δ)-DP algorithm $\mathcal{M}'$ for isotonic regression for ℓ on $\mathcal{X}_{\mathrm{grid}(w,h)}$ where $\delta \leq 0.01\varepsilon/D$. We use this algorithm to build a $(\ln(D/2), 0.25)$-DP algorithm $\mathcal{M}$ for privatizing a vector $\mathbf{z} \in [D]^m$ as follows:

- Create a dataset D that contains:
  - For all $i \in [m]$, r copies of $((i,(w-i)(D+1)+z_i),0)$ and r copies of $((i,(w-i)(D+1)+z_i+1),1)$.
  - $n - 2rm$ copies of $((1,1),y)$.
- Run $\mathcal{M}'$ on instance D to get f.
- Output a vector $\mathbf{z}'$ where $z'_i = \max\{j \in [D] \mid f((i,(w-i)(D+1)+j)) \leq 1/2\}$. (For simplicity, when such j does not exist let $z'_i = 0$.)

By group privacy, $\mathcal{M}$ is $(\ln(D/2), 0.25)$-DP. Furthermore, $\mathcal{L}(f^*; D) = 0$ and the expected empirical excess risk of $\mathcal{M}'$ is

$$\begin{aligned}
&\mathbb{E}_{f\sim\mathcal{M}'(D)}[\mathcal{L}(f;D)] \\
&\geq \tfrac{r}{n}\textstyle\sum_{i\in[m]}\left(\ell(f(i,(w-i)(D+1)+z_i),0) + \ell(f(i,(w-i)(D+1)+z_i+1),1)\right) \\
&\geq \tfrac{r}{n}\textstyle\sum_{i\in[m]} g(1/2)\cdot\mathbf{1}[z'_i \neq z_i] = \tfrac{Rr}{n}\cdot\|\mathbf{z}-\mathbf{z}'\|_0,
\end{aligned}$$

which must be at least $\Omega(Rrm/n) = \Omega\left(R\cdot\min\left\{1, \frac{w\cdot\log(h/w)}{\varepsilon n}\right\}\right)$ by Lemma 9. □

5 Additional Related Work

(Non-private) isotonic regression has been well-studied in the statistics and machine learning literature. The one-dimensional (aka univariate) case has long history including [8, 45, 4, 43, 44, 12, 7, 34, 13, 14, 48]; for a general introduction, the reader is referred to [21]. Moreover, isotonic regression has been studied in higher dimensions [23, 26, 25], including the sparse setting [20], as well as in the online learning setting [28]. A related line of work studies learning neural networks under (partial) monotonicity constraints [2, 49, 29, 39].

There has been a rich body of work on DP machine learning, including DP empirical risk minimization (ERM), e.g., [10, 5, 47, 46]; however, to the best of our knowledge none of these can be applied to isotonic regression to obtain non-trivial guarantees.

Another line of work related to our setting is around privately learning threshold functions [6, 19, 9, 1, 27], as explained in Section 3.2. We leveraged this relation to prove our lower bound for totally ordered case.

6 Conclusions

In this paper we obtained new private algorithms for isotonic regression on posets and proved nearly matching lower bounds in terms of the expected empirical excess risk. Although our algorithms for totally ordered sets are efficient, our algorithm for general posets is not. Specifically, a trivial algorithm would run in time $\exp(\tilde{O}(\mathrm{width}(\mathcal{X})))$. It remains an interesting open question whether this can be sped up. To the best of our knowledge, this question does not seem to be well understood even for the non-private setting, as previous algorithmic works primarily have focused on the totally ordered case. This complexity question remains an interesting future direction for research.

References

[1] N. Alon, R. Livni, M. Malliaris, and S. Moran. Private PAC learning implies finite littlestone dimension. In *STOC*, pages 852–860, 2019.

[2] N. P. Archer and S. Wang. Application of the back propagation neural network algorithm with monotonicity constraints for two-group classification problems. *Dec. Sci.*, 24(1):60–75, 1993.

[3] M. Ayer, H. D. Brunk, G. M. Ewing, W. T. Reid, and E. Silverman. An empirical distribution function for sampling with incomplete information. *Ann. Math. Stat.*, pages 641–647, 1955.

[4] R. E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. D. Brunk. *Statistical Inference Under Order Restrictions*. John Wiley & Sons, 1973.

[5] R. Bassily, A. Smith, and A. Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *FOCS*, pages 464–473, 2014.

[6] A. Beimel, K. Nissim, and U. Stemmer. Private learning and sanitization: Pure vs. approximate differential privacy. *Theory Comput.*, 12(1):1–61, 2016.

[7] L. Birgé and P. Massart. Rates of convergence for minimum contrast estimators. *Prob. Theory Rel. Fields*, 97(1):113–150, 1993.

[8] H. D. Brunk. Maximum likelihood estimates of monotone parameters. *Ann. Math. Stat.*, pages 607–616, 1955.

[9] M. Bun, K. Nissim, U. Stemmer, and S. P. Vadhan. Differentially private release and learning of threshold functions. In *FOCS*, pages 634–649, 2015.

[10] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *JMLR*, 12(3), 2011.

[11] R. P. Dilworth. A decomposition theorem for partially ordered sets. *Ann. Math.*, 51(1):161–166, 1950.

[12] D. L. Donoho. Gelfand n-widths and the method of least squares. *Technical Report, University of California*, 1991.

[13] C. Durot. On the l_p-error of monotonicity constrained estimators. *Ann. Stat.*, 35(3):1080–1104, 2007.

[14] C. Durot. Monotone nonparametric regression with random design. *Math. Methods Stat.*, 17(4):327–341, 2008.

[15] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pages 486–503, 2006.

[16] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[17] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[18] R. L. Dykstra and T. Robertson. An algorithm for isotonic regression for two or more independent variables. *Ann. Stat.*, 10(3):708–716, 1982.

[19] V. Feldman and D. Xiao. Sample complexity bounds on differentially private learning via communication complexity. In *COLT*, volume 35, pages 1000–1019, 2014.

[20] D. Gamarnik and J. Gaudio. Sparse high-dimensional isotonic regression. *NeurIPS*, 2019.

[21] P. Groeneboom and G. Jongbloed. *Nonparametric Estimation under Shape Constraints*. Cambridge University Press, 2014.

[22] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *ICML*, pages 1321–1330, 2017.

[23] Q. Han, T. Wang, S. Chatterjee, and R. J. Samworth. Isotonic regression in general dimensions. *Ann. Stat.*, 47(5):2440–2471, 2019.

[24] M. Hardt and K. Talwar. On the geometry of differential privacy. In *STOC*, pages 705–714, 2010.

[25] S. M. Kakade, V. Kanade, O. Shamir, and A. Kalai. Efficient learning of generalized linear and single index models with isotonic regression. In *NeurIPS*, 2011.

[26] A. T. Kalai and R. Sastry. The isotron algorithm: High-dimensional isotonic regression. In *COLT*, 2009.

[27] H. Kaplan, K. Ligett, Y. Mansour, M. Naor, and U. Stemmer. Privately learning thresholds: Closing the exponential gap. In *COLT*, pages 2263–2285, 2020.

[28] W. Kotłowski, W. M. Koolen, and A. Malek. Online isotonic regression. In *COLT*, pages 1165–1189, 2016.

[29] X. Liu, X. Han, N. Zhang, and Q. Liu. Certified monotonic neural networks. *NeurIPS*, pages 15427–15438, 2020.

[30] R. Luss, S. Rosset, and M. Shahar. Efficient regularized isotonic regression with application to gene–gene interaction search. *Ann. Appl. Stat.*, 6(1):253–283, 2012.

[31] P. Manurangsi. Tight bounds for differentially private anonymized histograms. In *SOSA*, pages 203–213, 2022.

[32] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. *Commun. ACM*, 53(9):89–97, 2010.

[33] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[34] M. Meyer and M. Woodroofe. On the degrees of freedom in shape-restricted regression. *Ann. Stat.*, 28(4):1083–1104, 2000.

[35] L. Mirsky. A dual of Dilworth's decomposition theorem. *AMS*, 78(8):876–877, 1971.

[36] C. Radebaugh and U. Erlingsson. Introducing TensorFlow Privacy: Learning with Differential Privacy for Training Data, March 2019. blog.tensorflow.org.

[37] T. Robertson, F. T. Wright, and R. L. Dykstra. *Order restricted statistical inference*. John Wiley & Sons, 1988.

[38] M. J. Schell and B. Singh. The reduced monotonic regression method. *JASA*, 92(437):128–135, 1997.

[39] A. Sivaraman, G. Farnadi, T. Millstein, and G. Van den Broeck. Counterexample-guided learning of monotonic neural networks. In *NeurIPS*, pages 11936–11948, 2020.

[40] T. Steinke and J. R. Ullman. Between pure and approximate differential privacy. *J. Priv. Confidentiality*, 7(2), 2016.

[41] Q. F. Stout. Unimodal regression via prefix isotonic regression. *Comput. Stat. Data Anal.*, 53(2):289–297, 2008.

[42] D. Testuggine and I. Mironov. PyTorch Differential Privacy Series Part 1: DP-SGD Algorithm Explained, August 2020. medium.com.

[43] S. Van de Geer. Estimating a regression function. *Ann. Stat.*, pages 907–924, 1990.

[44] S. Van de Geer. Hellinger-consistency of certain nonparametric maximum likelihood estimators. *Ann. Stat.*, 21(1):14–44, 1993.

[45] C. van Eeden. *Testing and Estimating Ordered Parameters of Probability Distribution*. CWI, Amsterdam, 1958.

[46] D. Wang, C. Chen, and J. Xu. Differentially private empirical risk minimization with non-convex loss functions. In *ICML*, pages 6526–6535, 2019.

[47] D. Wang, M. Ye, and J. Xu. Differentially private empirical risk minimization revisited: Faster and more general. In *NIPS*, 2017.

[48] F. Yang and R. F. Barber. Contraction and uniform convergence of isotonic regression. *Elec. J. Stat.*, 13(1):646–677, 2019.

[49] S. You, D. Ding, K. Canini, J. Pfeifer, and M. Gupta. Deep lattice networks and partial monotonic functions. *NIPS*, 2017.

[50] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *KDD*, pages 694–699, 2002.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] *See Section 6.*
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] *Since this is a purely theoretical paper regarding private algorithms for well studied ML task of isotonic regression, we do not foresee any immediate potential negative impacts.*
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] *Including some proofs in the supplementary material.*
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]