Differential Privacy and Robust Statistics

Cynthia Dwork Jing Lei

November 14, 2008

Abstract

In the past several years a promising approach to private data analysis has emerged, based on the notion of *differential privacy*. Informally, this ensures that any outcome of an analysis is "roughly as likely" to occur independent of whether any individual opts in to, or to opts out of, the database. In consequence, the specific data of any one individual can never greatly affect the outcome of the analysis. General techniques for ensuring differential privacy have now been proposed, and many datamining tasks can be carried out in a differentially private fashion, frequently with very accurate results.

In this work we explore privacy-preserving parameter estimation. Privacy-preserving statistics would appear to be connected to *robust statistics*, the subfield of statistics that attempts to cope with several small errors due, for example, to rounding errors in measurements, as well as a few arbitrarily wild errors occurring, say, from data entry failures. In consequence, in a robust analysis the specific data for any one individual should not greatly affect the outcome of the analysis. It would therefore seem that robust statistical estimators, or procedures, might form a starting point for designing accurate differentially private statistical estimators, and the approach based on *influence functions* is particularly suggestive of differential privacy.

We report here on a successful attempt to instantiate this intuition. We obtain differentially private algorithms for estimating the data scale, median, α-trimmed mean, and linear regression coefficients. Our algorithms always ensure privacy. Under mild statistical assumptions they produce highly accurate outputs, with distortion vanishing in the size of the dataset; however, when the statistical assumptions fail the algorithm may halt with an output "No Reply."

Our algorithms follow a new paradigm for differentially private mechanisms, which we call Propose-Test-Release (PTR). We give general composition theorems for PTR mechanisms.

1 Introduction and Background

1.1 Differential Privacy

Over the last few years a new approach to privacy-preserving data analysis, based on *differential privacy* [8, 5], has born fruit [9, 2, 8, 1, 17, 16, 3, 14]. Intuitively, this notion says that any possible outcome of an analysis should be "almost" equally likely, independent of whether any individual opts in to, or opts out of, the data set. Still speaking intuitively, this ensures that (almost, and quantifiably) no risk is incurred by joining a statistical database.

The key result for our purposes says, informally, that when presented with a query T mapping databases to $\mathbf{R}^k$, differential privacy can be achieved by adding noise proportional to ΔT independently to each of the k outputs of T, where ΔT is the maximum, over all databases D, D' differing in a single row[1], of $||T(D) - T(D')||_1$. The quantity ΔT is called the L_1 *sensitivity* of T[2].

[1] *i.e.*, $||D - D'||_0 = 1$

[2] ΔT is sometimes referred to as the *global* sensitivity of T, emphasizing that it is a worst-case (over all possible pairs of neighboring data sets) measure.

The sensitivity of a sequence of queries is bounded by the sum of the sensitivities of the individual queries (triangle inequality), so we can handle multiple queries by viewing them as a single, higher sensitivity, query.

This has given rise to the development of low-sensitivity algorithms for various analytical tasks, such as k-means clustering, singular value decomposition, learning of association rules, mutually consistent contingency table release, private learning, generation of synthetic data sets, and several tasks in learning theory [2, 17, 1, 14, 3]. Two remarks are in order.

1. The design of insensitive algorithms can require considerable re-thinking of existing algorithms.

2. Sometimes the outputs will be of low quality. For example, when running the perceptron algorithm, whether the privacy preserving version in [2] or the standard version, this occurs whenever no good hyperplane separator exists for the data set in question. In this case there is no reason to actually produce a separator, and a privacy-preserving algorithm that announced this fact and failed to produce any other output would not be doing a disservice.

1.2 Statistics and Robustness

In this work we turn to statistics, specifically, parameter estimation, complementing the previous work on datamining algorithms. The question of sensitivity is a natural one in statistics. The branch of statistics that focuses on insensitivity to outliers and small errors in data measurement is *robust statistics*. As we next explain, robust statistics is to statistics what agnostic learning is to learning.

A statistic is a quantity computed from a sample (the data set). Much of statistics assumes the samples are drawn i.i.d. from a distribution F in a family of distributions $\mathcal{F}$, and tries to characterize the distribution. For concreteness, think of $\mathcal{F}$ as the family of all one-dimensional normal distributions of the form $\mathcal{N}(\theta, 1)$; the goal is to find the mean θ. In contrast, robust statistics recognizes that in real life even the "best" distribution in $\mathcal{F}$ is only an approximation to the underlying distribution; that is, real life provides at best an approximation to a distribution in $\mathcal{F}$.

A classical result in statistics shows that the sample mean is the most *efficient* estimator of the mean of a normal distribution, under these ideal conditions, signifying that this estimator converges more quickly than any other as the number of samples increases [10]. On the other hand, a single very wild data point can move the sample mean arbitrarily. A much more robust estimator of location is the sample median, which is a better choice when the samples may come from a distribution that is only close to a normal, or perhaps not normal at all. Moreover, in our setting statistical efficiency is not a concern, as data are plentiful (we do, however, care about computational efficiency).

More generally, a robust procedure produces "essentially" the same output independent of the value of a (small number of) data points. Indeed, this is also the popular intuition: statistics about a population only are meaningful representations of properties of the population if the results robust to small changes in the set of people included in a study.

1.3 From Robustness to Privacy: Propose-Test-Release Algorithms

Our hope when beginning this investigation was to base the design of privacy-preserving algorithms for statistics on known robust procedures, allowing us to avoid the re-thinking of standard algorithms that was necessary in the data mining case (cf. [2]). The robust procedures that we

investigated were indeed excellent starting points, supporting the conjecture that a substantial amount of the re-thinking of classical estimators needed for ensuring privacy has already been done by the designers of robust statistics. We demonstrate this with procedures for finding data scale and location, and for linear regression (five algorithms in all).

One difficulty that we encountered is that in robust statistics the assumption is that there exists an underlying distribution that is "close to" the distribution from which the data are drawn, that is, that the real life distribution is a contamination of a "nice" underlying distribution, and that mutual dependence is limited. The resulting claims of insensitivity (robustness) are therefore probabilistic in nature even when the data are drawn iid from the nicest possible distribution. On the other hand, to apply the results of [8], calibrating noise to sensitivity in order to achieve differential privacy even in the worst case, we must cope with worst-case sensitivity. We address this by including explicit, differentially private, tests of the sensitivity of our computations on the given data set. A little more precisely, the algorithms propose a bound on sensitivity, either working with a default proposal or obtaining the proposal by engaging in preliminary differentially private computations; they then test the adequacy of the proposed bound, again in a privacy-preserving fashion. If the responses indicate high sensitivity, the algorithm halts. Since this decision is made based on the outcome of a differentially private test, no information is leaked by the decision itself. If the responses indicate the proposed bound is adequate, then the quantity is computed and noise is added according to a Laplace distribution with parameter depending on the proposal. For obvious reasons we call this the *Propose-Test-Release* paradigm.

In this paper, the proofs that our algorithms give privacy will generally be straightforward. The challenge will be to understand when they also give utility. For this, we rely on robust statistics, basing our algorithms on robust estimators. Roughly speaking, the proof of robustness involves showing that if the distribution has certain nice properties then the effect of adding a single *arbitrary* data point to a data set consisting of n iid samples from the distribution has, with overwhelming probability over the samples, a vanishing impact on the statistic. The quantification of "vanishing" will yield our bound on sensitivity; the statistical nature of the claim ("overwhelming probability over the samples") will give us utility in the statistical setting.

High sensitivity of an estimator for a given data set may be an indication that the statistic in question is not informative for the given data set, as in the bad cases for the perceptron algorithm, and there is no point in insisting on an outcome. As an example, suppose we are seeking the inter-quartile range of a data set in order to estimate scale. If the computed inter-quartile range is highly sensitive, for example, if deleting a single sample would double the statistic, then it is not an interesting statistic for this data set – it is certainly not telling us about the data set as a whole – and there is little point in pressing for an answer.

This observation has proved very powerful; for example, it has allowed us to obtain the first privacy-protective algorithm for finding the median when the range of the data is not known in advance. To our knowledge, ours are the first privacy-preserving algorithms for all the other tasks.

1.4 Statistica Estimators and The Influence Function

The material in this section is useful for understanding how we obtain the proposed bounds on sensitivity for the Propose-Test-Release paradigm, but it can be skipped without major impact on the ability to read the remainder of the paper.

Let T be a statistical estimator for θ; very roughly, this is a procedure that maps data samples to a real number or a vector of real numbers that approximates θ. We have in mind procedures such as the computation of the sample median or sample average. Let F^n denote the distribution on n-tuples obtained by taking n i.i.d. samples from random variable X whose distribution function

is F. Given n data points $D = \{x_1, \ldots, x_n\}$, the statistical estimator T can be viewed as a function on the set of data points: $T(x_1, \ldots, x_n)$, which is apparently random. However, most (reasonable) statistical estimators will converge to a non-random quantity as the sample size n tends to infinity. This limiting quantity depends only on the distribution F, denoted by $T(F)$. As a result, a statistical estimator can be viewed as a functional mapping the space of distribution functions to Euclidean space [12]. For example, suppose $X \in \mathbf{R}^1$ and $T(x_1, \ldots, x_n) = \sum_{i=1}^n x_i/n$ is the sample mean, then $T(F) = \int xf(x)dx = E_F X$, the expectation of F, given that $E_F X$ exists. Another example is the median: $T(x_1, \ldots, x_n) = x_{(\lfloor n/2 \rfloor)}$ (the $\lfloor n/2 \rfloor$th smallest data point), then $T(F) = F^{-1}(1/2)$, the median of the distribution, given that F has positive density at the median. In the asymptotic perspective, the robustness of T is defined, roughly, as the maximum of $|T(F) - T(G)|$, for a given F and for all G within a certain small distance of F. The intuition here is to ask "is it still estimating the same thing, or something close, even when the true unknown distribution F is a bit different from our assumption G?" Basing on such an intuition, a rigorous definition of (this asymptotic) robustness is through the concept of "influence function" (see Definition 2 and also [13, 12]).

In practice, it is often more interesting to look at the finite sample situation (the empirical data situation), so we first introduce the *empirical influence function*, denoted by $\mathrm{EIF}(x, T; D)$ which describes the effect on T of introducing an additional data point x.

Definition 1. The *empirical influence function* is given by

$$\mathrm{EIF}(x, T; \{x_1, \ldots, x_n\}) = (n+1)(T(x_1, \ldots, x_n, x) - T(x_1, \ldots, x_n)).$$

Since the empirical influence function depends on the observed data D, it is a random variable. If we are in a statistical setting, meaning that the data are actually i.i.d. samples from a distribution F, then a well-behaved empirical influence function should not depend too much on the specific set of samples, and a good understanding of the empirical influence function gives guidance regarding the expected local sensitivity.

To capture the statistical setting we are interested in $\sup_x \mathrm{EIF}(x, T; D)/(n+1)$ when the elements in D are drawn i.i.d. from the distribution F. To emphasize this, we will use the notation $|\sup_x \mathrm{EIF}(x, T; F^n)|$. For random data the local sensitivity is random, and local sensitivity is at least $1/(n+1) \times \sup_x |EIF(x, T, F^n)|$. This is a lower bound on the expected magnitude of the noise we need to add for privacy, using the techniques of [8] mentioned above. For example:

1. Let T be the sample mean. Then $|\sup_x \mathrm{EIF}(x, T; F^n)| = \infty$.

2. Let T be the sample median, and let F be a distribution with density 0 at the median $F^{-1}(\frac{1}{2})$. Then $|\sup_x \mathrm{EIF}(x, T; F^n)| = \Theta(n)$. This yields bounded local sensitivity in the statistical setting, independent of n.

3. Let T be the sample median, and let F be a distribution with positive density at $F^{-1}(\frac{1}{2})$. Then $|\sup_x \mathrm{EIF}(x, T; F^n)| = O_P(1)$. That is, the EIF converges to a distribution[3]. Thus, in the statistical setting, local sensitivity vanishes as n grows.

4. Let T be the α-trimmed mean, in which the top and bottom $\alpha/2$ fraction of the data points are discarded. Then $|\sup_x \mathrm{EIF}(x, T; F^n)| = O_P(1)$ if F has positive and continuous density on its support. Again, in the statistical setting, the local sensitivity vanishes.

The empirical influence function is typically approached via the *influence function* $\mathrm{IF}(x, T; F)$, an asymptotic notion describing how an estimator T applied to samples from F changes if we replace

[3]See Notation 8, in Section 2, for the definition of $O_P(1)$.

F by a distribution G with an infinitesimal contamination at x: $G = (1-t)F + t\Delta_x$, for very small t[4]. More precisely:

Definition 2. The *influence function* of T at F is given by

$$\mathrm{IF}(x, T; F) = \lim_{t \to 0} \frac{T((1-t)F + t\delta_x) - T(F)}{t}.$$

The *gross error sensitivity* of T and F is defined by

$$\gamma^*(T, F) = \sup_x |\mathrm{IF}(x, T; F)|.$$

The first principle in designing a robust estimator is to ensure that γ^* is bounded, as otherwise "an outlier might cause trouble" ([13]). By starting from estimators with bounded gross error sensitivity, we are able to ensure that, in addition to preserving privacy, our algorithms ensure excellent utility in the statistical setting.

1.5 Our Results

We investigate the use of robust estimators as starting points for privacy-preserving statistical procedures that obtain accurate outputs in the statistical setting. Our results are uniformly positive. Indeed, after obtaining privacy-preserving versions of the interquartile, median, and α-trimmed mean algorithms, we chose a very complicated analysis in the book *Robust Statistics* [12], which turned out to be of a regression algorithm of Hampel, and verified that this algorithm, too, quickly yielded a privacy-preserving variant. Along the way we found two other privacy-preserving estimators: a variant of the α-trimmed mean and a short-cut regression algorithm.

Our algorithms:

- always ensure privacy;
- ensure utility with high probability: any answer produced will, with high probability over the random choices made by the algorithm, and the choice of database (remember, we are interested in the statistical setting), be highly accurate – the distortion vanishes as n, the size of the data set, grows; and
- may produce the output "No Reply" ($\perp$) when the data fail to satisfy statistical assumptions.

Our notion of privacy is (ε, δ)-*differential privacy with negligible* δ, defined in Section 2.

Very roughly, our approach is to first propose a bound on the local sensitivity, and then test in a privacy-preserving fashion if the bound is sufficiently high, and, if so, to release the quantity of interest with noise calibrated to the proposed bound. Nissim, Raskhodnikova, and Smith were the first to exploit low local sensitivity to improve accuracy in favorable cases [17][5]. They demonstrated an insensitive method of upper bounding the local sensitivity, and then add noise calibrated to the computed bound. The theory they develop is inspiring, although their fully general techniques can be difficult to work with. In contrast, our approach is "quick and dirty:" our "No Reply" option considerably simplifies the algorithms. Intuitively, we are optimizing for the statistical setting.

[4] One technical difficulty is that the EIF does not always converge to the IF as n tends to infinity. Still, the intuition described here is valid.

[5] Given a data set D and a function f, Nissim *et al.* define the local sensitivity of f at D to be $\sup_{\{D' : |D - D'|_0 = 1\}} |f(D) - f(D')|$.

Indeed, it is the statistical setting that enables us to propose a realistic bound on the variability of the estimator, and we only require utility in this setting.

In more detail, we considered three estimation problems: data scale, data location, and linear regression. The classical estimators are, respectively, standard deviation, mean, and least squares regression, and their robust counterparts are, respectively, the interquartile range, the median (also the α-trimmed mean), and Hampel's most B-robust estimator (an L_1 optimization that; we denote by Algorithm $\mathbf{H}$)[6].

The robust estimators have the property that, for any given distribution F satisfying certain mild assumptions, with overwhelming probability over the choice of the database drawn from F^n, the local sensitivity is a random variable $g(n)$ such that $ng(n) \xrightarrow{d} f$, where f is random and its distribution depends only on the unknown underlying distribution F, independent of n. We need to test sensitivity because of the randomness of the local sensitivity and because the model might be incorrect, *i.e.*, the data may not be drawn from F, or the draws may be bizarre.

For the scale algorithm, $\mathcal{S}$, we test how far the database is from one with a sufficiently different interquartile distance (the difference obtained by lopping off the top and bottom quartiles and finding the spread of the remaining data points). If sufficiently far, then the algorithm proceeds; otherwise it halts with output $\perp$. Certain technical questions arise: first, we work with the logarithm of the scale, as our definition of "sufficiently different" is multiplicative. This is necessary, since an additive notion for difference of scale makes no sense – what would be the right scale for the additive amount? Second, for reasons related to privacy, we phrase the query by first fixing a discretization of the real line (independent of the data set), and then asking how much the dataset has to change in order to move the log of the interquartile range to a different bin in the discretization. Rather than respond to the question exactly, noise is added in keeping with the results of [8], so that the response is generated in a differentially private fashion (this will be the case for all the "how many points" questions in our informal descriptions here; we won't repeat this.) The test is problematic if the log of the interquartile range is very close to the boudary of its bin, so if an unfavorable reply is obtained we repeat the test with a discretization having bins of the same width, but "shifted" by half a bin (this suffices). The noise added to the logarithm of the interquartile range is determined by a Laplacian with parameter corresponding to the bin width the discretization. Again, in this algorithm the bin width is independent of the dataset, and the "propose" part of the PTR paradigm is just this fixed width.

The differentially private version of the α-trimmed mean is obtained easily from Algorithm $\mathcal{S}$, modified to find the α interquantile range; we defer further discussion to Section 5.

The median algorithm $\mathcal{M}$ has a scale input s which might be empty. If the scale is empty, then the algorithm computes a value for s Algorithm $\mathcal{S}$ just described. If Algorithm $\mathcal{S}$ returns $\perp$ then we output $\perp$. Otherwise, we discretize the line with bins whose width depend on the scale, and ask how much the database must change in order to drag the median out of its current bin. If the answer is too small, the algorithm outputs $\perp$; otherwise, noise is added to the median again according to a Laplacian with parameter corresponding to the width of a bin (now a function of the scale). For similar reasons to the previous case, if the first discretization yields $\perp$ we repeat the test for sensitivity of the median using a shifted discretization before producing $\perp$.

Both algorithms require careful but fairly straightforward privacy and utility analyses.

The short-cut regression algorithm, $\mathcal{R}_S$, similar to an algorithm proposed by Siegel (see Section 6.4.1 of [12]), and also reminiscent of the Subsample-and-Aggregate framework of Nissim,

[6]The "B" refers to *bias*; see [12], p. 87. The bias of an estimator for a quantity is the difference in expectation between the expected value for this quantity (the expectation is taken over the sample) and the expected output of the estimator (the expectation is taken over the sample and any randomness introduced by the estimation procedure).

Raskhodnikova, and Smith [17]. Here, we briefly describe the case in which the data points are in the plane and we are seeking a line, specified by 2 parameters $\beta = (\beta_1, \beta_2)$, that describes the dataset (the algorithm works in general dimension). Assume 2 divides n. The algorithm first randomly partitions the n inputs into disjoint blocks of 2 data points each. An approximation to (both coordinates of) β is computed from each block. This gives $n/2$ independent approximations to β. For each coordinate of β, run Algorithm $\mathcal{M}$ on a dataset consisting of the $n/2$ different values obtained for this coordinate to obtain a single output value for this coordinate. Assuming no invocation of Algorithm $\mathcal{M}$ results in $\perp$, output the vector consisting of these 2 coordinates.

Arguing utility for the short-cut algorithm is straightforward. However, the privacy argument is a little different due to the random partitioning and the aggregations of the $n/2$ independent estimates for β. Rather than give a special privacy proof for this algorithm we defer the proof of privacy to the section on (ϵ, δ)-PTR computations, where we explain how to apply the general composition results of that section to this algorithm.

Finally, we come to Algorithm $\mathcal{R}_H$, a differentially private version of the **H** regression algorithm. We examined algorithm **H** for two reasons: first, the complexity of its robustness analysis suggested in would be a good test case of our thesis that robust estimators are good starting points for privacy-preserving estimators. Second, it is more efficient, in the statistical sense (efficiency captures the question of how large n must be in order for the error bars on the estimate to be small), than the short-cut algorithm under certain circumstances[7].

This is by far our technically most challenging result, as the proofs of the running time and of utility in the statistical setting are complex.

The algorithm itself is not so hard to describe. Again, we focus in this high-level view on the case on the dimension 2 case. First we compute $\beta^*(D) = \mathbf{H}(D)$, without privacy (this is not released). As in the short-cut regression: the points are randomly partitioned into $n/2$ groups and an estimate for $\beta = (\beta_1, \beta_2)$ is obtained within each group. Now, rather than run the median algorithm on the estimates for each of β_1 and β_2, we run the scale algorithm, once to determine the scale of estimates for β_1 and once to determine the scale for β_2. We use these scales to determine four two-dimensional axis-parallel discretizations of the plane (a basic one and three shifts, obtained by shifting either one or both of the axis-parallel one-dimensional discretizations). We discuss the processing on a single discretization; as with algorithms $\mathcal{S}$ and $\mathcal{M}$, the remaining three need to be invoked only if our efforts with the first one return $\perp$; in general we try them in turn until a non-$\perp$ response is obtained. Note that the optimization problem might have multiple solutions, and we cannot specify $\beta^*(D)$ without knowing the exact optimization algorithm. Instead, we ask about the solution *set* of the optimization problem, denoted $B(D)$. Assume $B(D)$ is completely covered by one of cells of the discretization (we prove this to be the case whp in the statistical setting). We ask how many points need to be added to or deleted from D in order to get a database $\hat{D}$, such that $B(\hat{D})$ is not covered by the cell containing $B(D)$. If the answer is at least 2, then modifying the value of a single data point will not drag any part of $B(\hat{D})$ out of the current cell, which implies $\beta^*(D')$ will be in the same cell for all D' adjacent to D. Note that if $B(D) \not\subseteq C(D)$, where $C(D)$ is the set containing $\beta^*(D)$, then the answer to the question is 0.

Assuming a favorable response to the "How many points..." question, we will eventually add noise to each coordinate of $\beta^*(D)$ according to a Laplacian with parameter equal to the length of the corresponding side of the cell.

Answering the question of how many points need to be added or deleted before the solution set of the resulting database is no longer contained in the same cell as the cell containing $B(D)$ is

[7]For example, in the regression model defined in Section 7, when ϕ is standard normally distributed, and $E_F XX^T/||X||$ is diagonal.

not straightforward. Our starting point is the structure of the objective function f_D in the robust estimator **H** (see Equation (23)). First the convexity of f_D enables one to approach the minimum from the subdifferential, which can be computed easily from the data. On the other hand, f_D is piece-wise linear, so that computing the answer of the testing query reduces to evaluating the subdifferential of f_D at a poylnomial (in n) number of points, where the degree of the polynomial depends on the dimension. To carry out the computation it is sufficient to keep track of the intersections and line segments.

The proof of utility resembles the median case. Under statistical assumptions the empirical distribution is not far from the underlying distribution F. The main tool is simply law of large numbers and large deviation theory. The technical challenge is that one needs to bound the large deviation probability for infinitely many random variables simultaneously, (while for the median one only needs to worry about finitely many variables). The reason is that in one dimension the boundary of a bin is just two points, and in two dimensions, the boundary of a bin is a rectangle. We deal with this challenge using a tool from the study of empirical processes.

As for the privacy, in all cases the intuition is simple, but full proofs may not be, since some algorithms rely on the outputs of others – for example, both our regression and the location algorithms invoke our algorithm for scale. We have actually proved privacy twice. The first time involved brute force integration and conditional analysis. Noting large amounts of repetition in the proofs we abstracted the propose-test-release framework mentioned above and then proved composition theorems that captured the ways in which our algorithms were used in combination, specifically,

1. *Cascading*: running a sequence of protocols until a non-$\perp$ reply is obtained;
2. *Nested Computations*: using the output of one computation as input to another computation; and
3. *Parallel Composition*: running an algorithm multiple times, independently, for example in estimating data scale along multiple axes.

Our composition theorems show that our intuition for why privacy was preserved was correct and give tighter analyses than those available using other composition results in the literature.

1.6 Related Work

The most relevant related privacy results are the definitions of differential privacy [8, 5], its relaxation (ε, δ)-differential privacy [7], and the calibration of noise to sensitivity for maintaining privacy, already discussed [8].

Also mentioned above is the idea of calibrating noise to (something related to) local sensitivity, rather than global sensitivity [17].

In parallel with our efforts, Smith [19] investigated maximum likelihood estimators, showing that for well-behaved parametric probability models, one can construct an estimator whose distribution converges to that of the MLE. In particular, the estimator is efficient and asymptotically unbiased.

The relevant results from the statistics literature is the work on *influence functions* [13, 12] and the most B-robust regression estimator **H** [12].

2 Definitions

A database is a set of *rows*. We say databases D and D' *are adjacent* if they are of Hamming distance one[8]. In such cases we may may also say that D and D' are *neighbors*.

Definition 3. A randomized function $\mathcal{K}$ gives ε-*differential privacy* if for all nieghboring data sets D and D' and all $S \subseteq Range(\mathcal{K})$,

$$\Pr[\mathcal{K}(D) \in S] \quad \leq \quad \exp(\varepsilon) \times \Pr[\mathcal{K}(D') \in S]. \tag{1}$$

The probability is taken over the coin tosses of $\mathcal{K}$.

Being differentially private is a property of the mechanism $\mathcal{K}$, and is orthogonal to the auxiliary information available to the adversary/user and its computational power. Differential privacy is also an *ad omnia* (rather than *ad hoc*) guarantee, and addresses any concerns that an individual might have about allowing her data to be included in a database (see [6] for further discussion of this point).

The key result for differential privacy is due to Dwork, McSherry, Nissim, and Smith [8]. For this, we require some definitions.

Definition 4. For $f : \mathcal{D} \to R^d$, the L_1-*sensitivity* of f is

$$\Delta f \quad = \quad \max_{D,D'} \|f(D) - f(D')\|_1 \tag{2}$$

for all neighboring D, D'.

For many types of queries the sensitivity Δf will be quite small. In particular, the query "How far is the dataset from a dataset with property P?" has sensitivity 1.

The scaled symmetric exponential distribution with standard deviation $\sqrt{2}\Delta f/\varepsilon$ denoted $\mathrm{Lap}(\Delta f/\varepsilon)$, has mass at x proportional to $\exp(-|x|(\varepsilon/\Delta f))$. More precisely, let $b = \Delta f/\varepsilon$. The probability density function is $p(x) = \exp(-|x|/b)/2b$ and the cumulative distribution function is $D(x) = (1/2)(1 + \mathrm{sgn}(x)(1 - \exp(|x|/b)))$.

Theorem 5 ([8])**.** *Let $\mathcal{D}$ denote the universe of databases. For $f : \mathcal{D} \to R^k$, the mechanism $\mathcal{K}_f$ that on input a database DB computes $f(DB)$ and then adds independently generated noise with distribution $Lap(\Delta f/\varepsilon)$ to each of the k output terms and outputs these k sums, enjoys ε-differential privacy.*

Note that decreasing ε, a publicly known parameter, flattens out the $\mathrm{Lap}(\Delta f/\varepsilon)$ curve, yielding larger expected noise magnitude. When ε is fixed, functions f with high sensitivity yield flatter curves, again yielding higher expected noise magnitudes.

In this work we use the following relaxation of differential privacy.

Definition 6. A randomized function $\mathcal{K}$ gives (ε, δ)-*differential privacy* if for all data sets D and D' differing on at most one element, and all $S \subseteq Range(\mathcal{K})$,

$$\Pr[\mathcal{K}(D) \in S] \quad \leq \quad \exp(\varepsilon) \times \Pr[\mathcal{K}(D') \in S] + \delta \tag{3}$$

[8]In the literature, the definition is frequently slightly different: D and D' are adjacent, or differ in at most one element, if one is a proper subset of the other and the larger database contains just one additional row. The difference is generally insignificant and we work with whichever definition simplifies the task at hand. In our analyses it is frequently convenient to assume that the size of the database is known, so we use the "wiggling" version. The intuition for the definition is that now the adversary cannot tell if a member of the dataset has left and been replaced with a comlpetely different person.

The probability is taken is over the coin tosses of $\mathcal{K}$.

In this work we *always* have $\delta = \delta_n \in \nu(n)$, that is, δ_n grows more slowly than the inverse of any polynomial in the database size. Following common parlance, we say that δ *is negligible*.

We briefly explain the principal way in which we employ this relaxation. Recall that in the Propose-Test-Release framework, we first propose an upper bound on the local sensitivity, then test to see if the bound suffices; only if the bound suffices do we continue with the release of the statistic. For a given candidate bound B on the local sensitivity, let S_B be a (not necessarily proper) subset of the set of databases with local sensitivity at most B. Let D be the specific database. We ask how far D is from the set S_B, meaning, how many rows must be added to and deleted from D to obtain a database in S_B. The set S_B will be chosen to have the property that if D and D' are neighbors, then the difference in answer to this question is at most 1; in other words, we ensure that the query about distance to S_B is of sensitivity 1. We use the differentially private mechanism of [8] to respond to this question, say, adding noise according to the distribution $\text{Lap}(1/\varepsilon)$ for fixed, constant, ε. If the (differentially private) response is small, we do not proceed; however, if the response is large, say, at least $1 + \log^2 n$, we do proceed. The reasoning is that unless the coin flips of the privacy mechanism were very unlucky, and the draw from $\text{Lap}(1/\varepsilon)$ is at least $\log^2 n$ – an event that occurs with negligible probability $\nu(n)$ – B is indeed an upper bound on the local sensitivity.

Note that even the decision of whether or not to proceed preserves privacy. A careful conditional analysis thus yields (ε, δ)-differential privacy: for all D, D' differing in at most one element, for all possible subsets C of "No Response" union with the range of the estimator, the probability of obtaining an output in C when the database is D is at most δ plus e^{ε} times the probability of an output in C when the database is D', where $\delta \in \nu(n)$.

Intuitively, the only thing that can damage privacy is an unlucky coin flip sequence in the noise generation process when responding to the test for sufficiency of the proposed noise magnitude, and this occurs with only negligible probability. In reasoning about protocol composition, we wish to factor out these unlikely events, arguing that "with overwhelming probability" we have differential privacy. Strictly speaking, this is meaningless; the only formal guarantee we offer is (ε, δ)-differential privacy. Nonetheless, the intuition helped in designing the protocols, and the composition theorems for Propose-Test-Release algorithms capture turn the intuition into the rigorous guarantee.

3 Additional Notation

We first declare some notations and terms used throughout the following discussion.

1. D and D': a pair of adjacent databases.
2. $\mathcal{D}$: the space of all databases.
3. $\mathcal{C}$ (also $\mathcal{C}'$): some general measurable space containing the range of the query function. Usually we can think it as $\mathbf{R}^d$ for some integer d, e.g., in our examples of scale, median, regression, etc. Note that the notation $\mathcal{C}$ ($\mathcal{C}'$) may refer to different spaces in different expressions.
4. n: the size of database D, which can be released privately with high accuracy [8]. Here for presentation simplicity we treat it as a known constant.
5. "change a data point" means modifying the value of a data point.
6. We do not worry about measurability. That is, for the sets considered in this paper we always assume they are measurable in the corresponding probability space.

7. Many of the analyses and statements in this work are probability-theoretic. In our study there are two sources of randomness.

   (a) The first is the coin flips made by the (random) algorithms, typically, the generation of Laplacian random variables. In these algorithms there is always an input database. We usually want to compare the probabilities of the same event but with adjacent input databases D and D', for example, compare $\text{Prob}(\mathcal{T}(D) \in C|D)$ and $\text{Prob}(\mathcal{T}(D') \in C|D')$. In such a comparison we always condition on D and D'. In other words, the databases are always considered as non-random. We use the convention $P(\cdot)$ to denote the probability of a certain random event in the algorithm when the input database is D, while $P'(\cdot)$ refers to the probability when the input database is D'. Similarly $p(X = x)$ ($p'(X = x)$) denotes the probability density of random variable X at x, with the input database D (D'); for example, we might have $X = \mathcal{T}(D)$. Here, again, the randomness is provided by the algorithm and the database is considered to be fixed. Note that sometimes the random variable has both a continuous part and a discrete part; then $p(X = x)$ denotes the density if x is in the continuous part and the probability mass if x is in the discrete part. In particular, when $\mathcal{T} : \mathcal{D} \to \mathbf{R} \cup \{\perp\}$, $p(\mathcal{T}(D) = 3.14159)$ is an example of the continuous part, and $p(\mathcal{T}(D) = \perp)$ is an example of the discrete part. In the first case the expression denotes density, in the second it denotes probability mass. Our statement about privacy will be in terms of $P(\cdot)$ and $P'(\cdot)$.

   (b) The second source of randomness is the randomness in creating the database. That is, when we assume the database consists of independent random samples from a underlying distribution F, the whole data set D is random. We use $P_F(\cdot)$ to denote the probability of the random event of this type. We also use $E_F(\cdot)$ to denote expectation taken over choice of a database consisting of independent random samples from an underlying distribution F.

   (c) $\tilde{P}(\cdot)$ refers to the probability considering both sources of randomness. Our statement about utility will be in terms of $\tilde{P}(\cdot)$.

8. Let a_n be a (random) sequence. We say $a_n = O_P(1)$ if for any $\epsilon > 0$, there exists M, such that $P(|a_n| > M) < \epsilon$ for all n. In addition, if a_n, b_n are two random sequences we say $a_n = O_P(b_n)$ if $a_n/b_n = O_P(1)$.

4 The scale

The interquartile range (IQR) ([11]) is a well-known robust estimate for the scale (dispersion) of the data, and is used in applications such as histogram construction[9]. We give a simple algorithm for differentially private release of interquartile range. The main idea of our algorithm is to propose a magnitude of noise for reporting the interquartile range and then test, privately, whether such a magnitude suffices for privacy. If the test succeeds, which we prove will be the typical case in the statistical setting, then the noisy interquartile range is released; otherwise the algorithm either returns "$\perp$" (No Response) or suggests another noise magnitude.

Before going into details, consider the following rough intuition. Suppose the data are i.i.d. samples drawn from an underlying distribution F. Then $\text{IQR}(F)$ may be defined as $F^{-1}(3/4) - F^{-1}(1/4)$; this is a constant, depending only on F. It might be very large, or very tiny, but either way, if the density of F is sufficiently large at the two quartiles, then given enough samples from

[9]The interquartile range is the difference between the $3n/4$th and $(n/4+1)$st order statistics.

F the sample interquartile distance should be close to $\mathrm{IQR}(F)$. Still speaking intuitively, if the sample interquartile range is very sensitive, for example, can double or halve if we change one data point, then either we have been very unlucky with the samples or the data don't come from a "nice" distribution. Either way it makes sense to abort the procedure and output "$\perp$." On the other hand if, as in the typical case, the sample interquartile cannot change by a factor of 2 by modifying a single point, then the logarithm base 2 of the sample interquartile has local sensitivity bounded by 1. This lets us release an approximation to the logarithm by adding to the logarithm noise with distribution $\mathrm{Lap}(1/\varepsilon)$ Alternatively, we can compute a noisy version y of this logarithm and release 2^y. Finally, we must ensure that the decision of whether or not to reply is not in itself disclosive. This is done via query $\mathbf{Q_0}$ below. Finally, to ensure utility, we do not use 2 for the base of the logarithm; rather, we use $1 + 1/\ln n$.

We first explain our choice of base. If the data are independently drawn from an underlying distribution F, then the deviation of the sample interquartile range $\mathrm{IQR}(D)$ from $\mathrm{IQR}(F)$ is $O_{P_F}(\frac{1}{\sqrt{n}})$, and also $\ln(\mathrm{IQR}(D)) - \ln(\mathrm{IQR}(F)) = O_{P_F}(\frac{1}{\sqrt{n}})$, where $\mathrm{IQR}(D)$ and $\mathrm{IQR}(F)$ denote the sample interquartile range and the interquartile range of the underlying distribution, respectively.[10] Intuitively, in order to achieve differential privacy, the proposed magnitude of the additive noise for the logarithm of sample interquartile range must be large enough to dominate $\frac{1}{\sqrt{n}}$, the deviation term $\ln(\mathrm{IQR}(D)) - \ln(\mathrm{IQR}(F))$. On the other hand, for the sake of good utility the noise should be small enough to give accurate release, From this perspective $\frac{1}{\ln n} > \frac{1}{\sqrt{n}}$ seems a good choice. We use $\ln(1 + 1/\ln n)$, which is close to $1/\ln n$ but which makes the calculation easier.[11]

To test whether the magnitude of noise is sufficient for differential privacy, we *discretize* $\mathbf{R}^1$ into disjoint bins $\{[kw_n, (k+1)w_n)\}_{k\in\mathbf{Z}}$, where the interval length $w_n = \ln(1 + 1/\ln n)$. Note that looking at $\ln(IQR(D))$ on the scale of w_n is equivalent to looking at $\log_{1+\frac{1}{\ln n}}(IQR(D))$ on the scale of 1, and here the scaled bins are just intervals whose endpoints are a pair of adjacent integers: $B_k^{(1)} = [k, k+1),\ k \in \mathbf{Z}$. Let $H_n(D) = \log_{1+\frac{1}{\ln n}}(IQR(D))$. Then we can find k_1 such that $H_n(D) \in [k_1, k_1 + 1)$. Consider the following testing query:

> $\mathbf{Q_0}$: How many data points need to change in order to get a new database $\hat{D}$ such that $H_n(\hat{D}) \notin B_{k_1}^{(1)}$?

Denote the answer to Q_0 by $A_0(D)$. If $A_0(D) \geq 2$, then all D' adjacent to D satisfy $|H_n(D') - H_n(D)| \leq 1$. As a result, it is sufficient to add noise with $\mathrm{Lap}(1/\varepsilon)$ distribution to $H_n(D)$. However, to ensure privacy, the algorithm uses $R_0(D) = A_0(D) + z_0$ instead of $A_0(D)$, where $z_0 \sim \mathrm{Lap}(1/\varepsilon)$. The algorithm releases $H_n(D)$ with noise calibrated to $1/\varepsilon$ for large values of R_0, and returns $\perp$ (no response) for small values of $R_0(D)$.

There is one remaining problem. If $H_n(D)$ lies close to an integer, i.e., $H_n(D)$ is close to one of the endpoints of the interval $[k_1, k_1 + 1)$, the algorithm is likely to return $\perp$. This problem could be avoided by also considering a second discretization $\{B_k^{(2)} = [k - 0.5, k + 0.5)\}_{k\in\mathbf{Z}}$. We denote the two discretizations by $B^{(1)}$ and $B^{(2)}$ respectively.

We now present the full algorithm.

[10] Consider $\ln(x)$ as a function of x. Its derivative is $1/x$. Then for y close to x we have, $\ln(y) - \ln(x) = (y - x)/x + o(y - x)$. Now take $x = \mathrm{IQR}(F)$ and $y = \mathrm{IQR}(D)$.

[11] Actually $\Omega(n^{-1/2+\delta})$ with some small $\delta > 0$ would work, here we just explain the feasibility but not focus on the optimality of the magnitude of noise.

The algorithm $\mathcal{S}$:

- Input: (D, n, ε).

1. For the jth discretization ($j = 1, 2$)
   - Compute $R_0(D) = A_0(D) + z_0$, where z_0 is a random draw from $\text{Lap}(1/\varepsilon)$.[a]
   - If $R_0 \leq \ln^2 n + 1$, let $s^{(j)} = \perp$.
   - Otherwise let $s^{(j)} = IQR(D) \times (1 + \frac{1}{\ln n})^{z_s^{(j)}}$, where $z_s^{(j)}$ is another random draw from $\text{Lap}(1/\varepsilon)$.
2. If $s^{(1)} \neq \perp$, return $s^{(1)}$.
   Otherwise return $s^{(2)}$.

[a] $IQR(D) = 0$ is fine, since one can define $\log 0 = -\infty$, $\lfloor -\infty \rfloor = -\infty$, and let $[-\infty, -\infty) = \{-\infty\}$.

Note that the algorithm can be optimized by only computing $s^{(2)}$ if $s^{(1)} = \perp$. For technical reasons in Section 8 it is easier to write the unoptimized version here. We have the following theorem describing the privacy and utility of the algorithm $\mathcal{S}$:

Theorem 7. *(a) The algorithm $\mathcal{S}$ is $(4\varepsilon, n^{-\varepsilon \ln n})$-differentially private.*

(b) The computation for $\mathcal{S}$ is $O(n)$ assuming the data are sorted.

(c) If

$$D = (X_1, ..., X_n), \qquad X_i \overset{iid}{\sim} F \tag{4}$$

where F is differentiable with positive derivatives at both the lower and upper quartiles, then

$$\tilde{P}(\mathcal{S}(D) = \perp) = O(n^{-\varepsilon \ln n}),$$

and

$$\mathcal{S}(D) - IQR(F) \xrightarrow{\tilde{P}} 0.$$

(d) Under the same conditions as in (c), for any $\alpha > 0$,

$$P\left(\mathcal{S}(D) \in [n^{-\alpha} IQR(D), n^{\alpha} IQR(D)]\right) \geq 1 - O(n^{-\alpha\varepsilon \ln n}).$$

As a result, we have

$$\tilde{P}\left(\mathcal{S}(D) \in \left[\frac{1}{2} n^{-\alpha} IQR(F), 2n^{\alpha} IQR(F)\right]\right) \geq 1 - O(n^{-\alpha\varepsilon \ln n}).$$

In Theorem 7, (a) and (b) ensure, respectively, the privacy and ease of computation of $\mathcal{S}$, (c) says that when the data come from a nice distribution the algorithm gives meaningful output with high probability. and (d) ensures that the output of $\mathcal{S}$ will not be crazily absurd. Note that although Theorem 7(c) says that the error vanishes with n, the rate of convergence is slow ($1 - n^{-c}$ for some $c > 0$). Nonetheless, Algorithm $\mathcal{S}$ is quite powerful: using this algorithm as a subroutine to estimate scale allows us to devise the first differentially private median algorithm that does not require range information for the data.

Proof of theorem 7 We first focus on a single discretization, and omit the index (j), $j = 1, 2$.

Lemma 8. *The sensitivity of query Q_0 is at most 1.*

Proof. If $H_n(D)$ and $H_n(D')$ are in different bins, then $A_0(D) = A_0(D') = 1$. Otherwise, they are in the same bin, then $A_0(D) \leq A_0(D') + 1$ and $A_0(D) \leq A_0(D') + 1$. In both cases we have $|A_0(D) - A_0(D')| \leq 1$. □

If $A_0(D) \geq 2$, then changing a single data point can not move the logarithm of the interquartile range out of the current bin; therefore the change is at most the bin width. Thus, it suffices to calibrate the noise to the bin width. Here we use $\mathcal{D}_0$ to denote such nice databases:

Definition 9. $\mathcal{D}_0 = \{D : A_0(D) \geq 2\}$.

Then we have the following lemma.

Lemma 10. *Let s be shorthand for the result obtained with a single discretization.*

(a) $P(s = \perp) \leq e^{\varepsilon} P'(s = \perp)$.

(b) $P(s \neq \perp) \leq \frac{1}{2} n^{-\varepsilon \ln n}$ *for all* $D \notin \mathcal{D}_0$.

(c) $P(s \in C) \leq e^{2\varepsilon} P'(s \in C)$, *for all* $C \subseteq \mathbf{R}^+$ *and* $D \in \mathcal{D}_0$.

Proof. (a) Note that for all $D \in \mathcal{D}$, $s(D) = \perp \Leftrightarrow R_0(D) \leq \ln^2 n + 1$. The desired inequality follows by applying Theorem 5 on query Q_0 which is of (global) sensitivity at most 1 (by Lemma 8).

(b) When $D \notin \mathcal{D}_0$, then $A_0(D_1) \leq 1$. As a result

$$\begin{aligned} P(s \neq \perp) &\leq P(R_0 > \ln^2 n + 1) \\ &= P(A_0 + z_0 > \ln^2 n + 1) \\ &\leq P(z_0 \geq \ln^2(n)) \\ &\leq \frac{1}{2} n^{-\varepsilon \ln(n)}. \end{aligned}$$

(c) The assumption $D \in \mathcal{D}_0$ implies $A_0(D) \geq 2$, whence $|H_n(D) - H_n(D')| \leq 1$, so

$$\begin{aligned} & P(s \in C) \\ =& \int_{v \in C} \int_{u > \ln^2(n)+1} p(R_0 = u) p\left(H_n + z_s = \log_{1+\frac{1}{\ln n}}(v) \middle| R_0 = u \right) du dv \\ =& \int_{v \in C} \int_{u > \ln^2(n)+1} p(R_0 = u) p\left(z_s = \log_{1+\frac{1}{\ln n}}(v) - H_n \right) du dv \\ \leq& \int_{v \in C} \int_{u > \ln^2(n)+1} e^{\varepsilon} p'(R_0 = u) e^{\varepsilon} p'\left(z_s = \log_{1+\frac{1}{\ln n}}(v) - H_n \right) du dv \\ =& e^{2\varepsilon} P'(s \in C). \end{aligned}$$

Here the inequality holds because $|A_0(D_1) - A_0(D_2)| \leq 1$ and $|H_n(D_1) - H_n(D_2)| \leq 1$.

□

Lemma 10 (c) implies the following useful corollary on the density of the outcome $\mathcal{S}(D)$:

Corollary 11. *If $D \in \mathcal{D}_0$, then for all $s_0 \in \mathbf{R}^+$*

$$P(s \in ds_0) \le e^{2\varepsilon} P'(s \in ds_0).$$

It is clear that all the results in Lemma 8, Lemma 10 and Corollary 11 hold for both discretizations, so we can replace s by $s^{(j)}$ and $\mathcal{D}_0$ by $\mathcal{D}_0^{(j)}$ in those results with $j = 1, 2$.

Proof of theorem 7. (a) (Privacy.) First notice, by Lemma 10, that both $s^{(1)}$ and $s^{(2)}$ are $(2\varepsilon, \frac{1}{2}n^{-\varepsilon \ln n})$-differentially private. That is, for any $C^* \subseteq \mathbf{R}^{\ge 0} \cup \{\perp\}$, we have

$$P(s^{(j)} \in C^*) \le e^{2\varepsilon} P'(s^{(j)} \in C^*) + \frac{1}{2} n^{-\varepsilon \ln n}.$$

Then by independence it is straightforward to show that, using similarly the conditional argument in Lemma 10 (which also appears in the proof of Lemma 16), for any $C \subseteq \left(\mathbf{R}^{\ge 0} \cup \{\perp\}\right)$, we have

$$\begin{aligned}
&P\left(\mathcal{S} \in C\right) \\
\le& P(s^{(1)} \in C \backslash \{\perp\}) + P(s^{(1)} = \perp, s^{(2)} \in C \backslash \{\perp\}) \\
&\quad + P(s^{(1)} \in C \cap \{\perp\}, s^{(2)} \in C \cap \{\perp\}) \\
\le& e^{2\varepsilon} P'(s^{(1)} \in C \backslash \{\perp\}) + \frac{1}{2} n^{-\varepsilon \ln n} + e^{3\varepsilon} P'(s^{(1)} = \perp, s^{(2)} \in C \backslash \{\perp\}) + \frac{1}{2} n^{-\varepsilon \ln n} \\
&\quad + e^{2\varepsilon} P'(s^{(1)} \in C \cap \{\perp\}, s^{(2)} \in C \cap \{\perp\}) \\
\le& e^{3\varepsilon} P'\left(\mathcal{S} \in C\right) + n^{-\varepsilon \ln n},
\end{aligned}$$

which indicates that $\mathcal{S}$ is $(3\varepsilon, n^{-\varepsilon \ln n})$-differentially private.

(b) (Ease of computation.) It suffices to show that A_0 can be computed in linear time. In fact, it is easy to check that one can compute $A_0(D)$ by considering $O(n)$ sliding intervals with width $(1 + \frac{1}{\ln n})^{k_1}$ and $(1 + \frac{1}{\ln n})^{k_1 + 1}$, and with one end point in D, while the computation for each interval is $O(1)$.

(c) (Good behavior in the statistical setting.) Let q_1 and q_2 be the lower and upper quartiles of F, respectively. Let $l_j = q_j - n^{-1/3}$, $r_j = q_j + n^{-1/3}$, $l'_j = q_j - 2n^{-1/3}$, $r'_j = q_j + 2n^{-1/3}$, for $j = 1, 2$, by differentiability and continuity of the derivatives, one can find constant $\xi > 0$ which depends only on F, such that for large enough n

1. $r'_1 < l'_2$.
2. $F'(x) > \xi$, for all $x \in [l'_1, r'_1] \cup [l'_2, r'_2]$.
3. $\xi n^{2/3} > 4 \ln^2 n + 4$.
4. $\left(\frac{r'_2 - l'_1}{l'_2 - r'_1}\right)^4 < 1 + \frac{1}{\ln n}$.

Intuitively, the last condition is not unreasonble, since $(\mathrm{IQR}(F) + 4n^{-1/3})/(\mathrm{IQR}(F) - 4n^{-1/3})$ is very close to 1; even more to the point it ensures that $\log_{1+1/\ln n}(\mathrm{IQR}(F) + 4n^{-1/3}) - \log_{1+1/\ln n}(\mathrm{IQR}(F) - 4n^{-1/3}) < 1/4$, whence the two logarithms will lie in the same bin in at least one of the discretizations. We consider the two following random events, where the randomness comes from the random draw of D from F, i.e. $P_F(\cdot)$: (denote the lower and upper sample quantiles by $q_1(D)$ and $q_2(D)$, respectively)

(i) $E_1 = \{q_1(D) \in (l_1, r_1), q_2(D) \in (l_2, r_2)\}$. We have for $j = 1, 2$

$$\begin{aligned} & P_F(q_j(D) \notin (l_j, r_j)) \\ =& P_F(q_j(D) \le l_j) + P(q_j(D) \ge r_j) \\ \le& 2P_F(\sup_x |F(x) - F_n(x)| \ge n^{-1/3}\xi) \\ \le& 2c_1 e^{-c_2 n^{1/3}}, \end{aligned}$$

the last inequality is a well known result about the deviations of the empirical distribution (see [15]), where c_1 and c_2 are numerical constants depending only on F, and they may take different value when they appear in different places throughout this paper.

Therefore we have

$$P_F(E_1) \ge 1 - c_1 e^{-c_2 n^{1/3}}. \tag{5}$$

(ii) Define $\rho := \min_j \min\left(|D \cap (l'_j, l_j)|, |D \cap (r_j, r'_j)|\right)$. The second event is $E_2 := \{\rho \ge \frac{1}{2}\xi n^{2/3}\}$. We first investigate $|D \cap (l'_1, l_1)|$. Define the Bernoulli random variable Y by

$$Y = \begin{cases} 1 & \text{if } X \in (l'_1, l_1), \\ 0 & \text{otherwise.} \end{cases}$$

By our second criterion in the condition on n, we have $P_F(Y = 1) \ge \xi n^{-1/3}$, then by Heoffding's inequailty

$$\begin{aligned} & P_F\left(|D \cap (l'_1, l_1)| < \frac{1}{2}\xi n^{2/3}\right) \\ \le& P_F\left(\left|\sum_{i=1}^n Y_i - nEY\right| \ge \frac{1}{2}\xi n^{2/3}\right) \\ \le& 2e^{-\xi^2 n^{1/3}/8}. \end{aligned}$$

Clearly the same argument applies to the other 3 intervals, therefore we have

$$P_F(E_2) \ge 1 - c_1 e^{-c_2 n^{1/3}}. \tag{6}$$

Now consider $H_n(F)$ and the intervals covering it. We say that a point is *well covered* by an interval if it is inside that interval and at least $\frac{1}{4}$ away from both endpoints. There are two discretizations, so there are two bins covering $H_n(F)$, one in each discretization, namely $B^{(1)}_{k_1}$ and $B^{(2)}_{k_2}$. By our construction of $B^{(1)}$ and $B^{(2)}$, at least one of $B^{(1)}_{k_1}$ and $B^{(2)}_{k_2}$ well covers $H_n(F)$.

Suppose $H_n(F)$ is well covered by an interval B. On event $E_1 \cap E_2$, if one changes at most $\frac{1}{2}\xi n^{2/3}$ data points, letting $\hat{D}$ be the resulting database we still have $q_j(\hat{D}) \in [l'_j, r'_j]$. Therefore on $E_1 \cap E_2$, $|H_n(\hat{D}) - H_n(F)| < \frac{1}{4}$ because of the fourth criterion in our conditions on n. Apparently $|H_n(D) - H_n(F)| < \frac{1}{4}$ on $E_1 \cap E_2$. So we have $H_n(D) \in B$ and $H_n(\hat{D}) \in B$, with probability at least $1 - c_1 e^{-c_2 n^{1/3}}$.

By our construction of $B^{(1)}_{k_1}$ and $B^{(2)}_{k_2}$, at least one of them well covers $H_n(F)$, as a result, for at least one discretization, we have $A_0(D) \ge 2\ln^2 n + 2$, with probability at least $1 - c_1 e^{-c_2 n^{-1/3}}$, then the corresponding s_j is not $\perp$ (note that $n^{-\varepsilon \ln n} = \omega(e^{-c_2 n^{1/3}})$ for any constant c_2).

Then we have the desired result:

$$\tilde{P}(\mathcal{S}(D) = \perp) = \tilde{P}(s_1(D) = \perp \text{ and } s_2(D) = \perp) = O(n^{-\varepsilon \ln n}). \tag{7}$$

The second claim, ie, that $\mathcal{S}(D) - IQR(F) \xrightarrow{\tilde{P}} 0$, follows from consistency of sample quantiles[12] and the fact that $(1 + \frac{1}{\ln n})^{z_s^{(j)}} \xrightarrow{P} 1$, for $j = 1, 2$.

(d) (Accuracy.)

$$\begin{aligned}
& P\left(\mathcal{S}(D) \in [n^{-\alpha} IQR(D), n^{\alpha} IQR(D)]\right) \\
\geq & 1 - P(\mathcal{S}(D) = \perp) - 2P\left(\left(1 + \frac{1}{\ln n}\right)^{z_s^{(1)}} \notin [n^{-\alpha}, n^{\alpha}]\right) \\
\geq & 1 - O(n^{-\varepsilon \ln n}) - 2P\left(|z_0^{(1)}| \ln\left(1 + \frac{1}{\ln n}\right) > \alpha \ln n\right) \\
\geq & 1 - O(n^{-\varepsilon \ln n}) - 2P\left(|z_0^{(1)}| > \alpha \ln^2 n\right) \\
= & 1 - O(n^{-\varepsilon \ln n}) - 2n^{-\alpha\varepsilon \ln n} \\
\geq & 1 - O(n^{-c_1 \ln n}).
\end{aligned}$$

The second statement follows from the fact that under the assumptions, on E_1,

$$\frac{1}{2} IQR(F) \leq IQR(D) \leq 2IQR(F).$$

□

5 α-Trimmed Mean

Let $D = (x_{(1)}, \ldots, x_{(n)})$ be an ordered data set, such that $x_{(i)} \in \mathbf{R}^1, x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(n)}$. For any $\alpha \in (0, 1)$, the α-trimmed mean is defined as

$$m_\alpha(D) = \frac{\sum_{i=\lceil n\alpha/2 \rceil + 1}^{\lfloor n(1-\alpha/2) \rfloor - 1} x_{(i)}}{\lfloor n(1 - \alpha/2) \rfloor - \lceil n\alpha/2 \rceil - 1}.$$

Then an (ϵ, δ)-differentially private algorithm for the α-trimmed mean follows almost immediately from the scale algorithm. First note that the scale algorithm can be used to compute any inter-quantile range of the data set by simply replacing the lower and upper quartiles by another pair of lower and upper quantiles.

Suppose $D' = (x_{(1)}, \ldots, x'_{(j)}, \ldots, x_{(n)})$ is an adjacent data set, with the value of $x_{(j)}$ modified to $x'_{(j)}$, in an arbitrary way. Then we have

$$|m_\alpha(D') - m_\alpha(D)| \leq \frac{x_{\lfloor n(1-\alpha/2) \rfloor} - x_{\lceil n\alpha/2 \rceil}}{(1 - \alpha)n - 2}. \tag{8}$$

Observe that the numerator on the right hand side of (8) is just the distance between the $\frac{\alpha}{2}$ and $1 - \frac{\alpha}{2}$ quantiles, denoted by $IQR_\alpha(D)$. So our algorithm for α-trimmed mean is, supposing $\kappa \in (0, 1)$

[12] Consistency says that $|IQR(D) - IQR(F)|$ converges to 0 in probability.

is some pre-chosen parameter,

The algorithm $\mathcal{TM}$

- **Input:** $(D, n, \varepsilon, \alpha)$.
- **Algorithm**
  1. Run algorithm $\mathcal{S}$ on (D, n, ε) computing the distance between lower and upper α quantiles. Denote the output by s_α.
  2. If $s_\alpha = \perp$, then return $\perp$. Otherwise return

$$m_\alpha(D) + \frac{s_\alpha n^\kappa z}{(1-\alpha)n - 2}$$

  where z is a random draw from Lap(1).

It follows from Theorem 7 (d) that with probability $1 - O(n^{-cn^{1/3}})$ we have $s_\alpha n^\kappa \geq IQR_\alpha(D)$, indicating the algorithm $\mathcal{TM}$ is $(3\varepsilon, \nu(n))$-differentially private, by a conditional argument similar as in Lemma 16.

On the other hand the algorithm $\mathcal{TM}$ enjoys nice utility, since it returns $\perp$ only if the algorithm $\mathcal{S}$ returns outputs $\perp$ which happens with $\nu(n)$ probability when the data set is "nice". Moreover, as we have shown that $\mathcal{S}(D)$ is a consistent estimator of $F^{-1}(1-\alpha/2) - F^{-1}(\alpha/2)$, the additive noise on the α-trimmed mean is of order $O(n^{-1+\kappa})$, which can be arbitrarily close to $O(n^{-1})$.

6 The median

We extend the idea used in the previous section to the sample median, where the output of the scale algorithm is used to construct the disretizations. In the previous section we used $\ln(1 + \frac{1}{\ln n})$ as the order of magnitude of additive noise for the natural logarithm of the interquartile range. However, for the median case, we need to incorporate in the magnitude of noise some quantity describing the "spread" of the distribution. The reason is easy to see in two ways: first, intuitively, the magnitude of noise should depend on how far the data points are from each other, so the dispersion of the data matters; second, theoretically, if the data are drawn from distribution F, the uncertainty (asymptotic variance) of the sample median is $1/2F'(m(F))$, i.e., the inverse of twice the density at the median, where $m(F)$ denotes the median of the distribution F. If we re-scale the distribution by a factor of h, then the density at median becomes $F'(m(F))/h$, consequently the asymptotic variance of the median becomes $h/2F'(m(F))$. So roughly speaking, the uncertainty of sample median is proportional to the scale of data, and it is natural to require that the magnitude of additive noise be proportional to the scale, of which the IQR is a good robust estimator ([11]).

Suppose the algorithm $\mathcal{S}$ returns a number $s \neq \perp$, we can discretize $\mathbf{R}^1$ in two ways: $B^{(1)} = \left\{B_k^{(1)} = [kh, (k+1)h)\right\}_{k\in\mathbf{Z}}$, and $B^{(2)} = \left\{B_k^{(2)} = [(k-0.5)h, (k+0.5)h)\right\}_{k\in\mathbf{Z}}$, where $h = h(s) = sn^{-1/3}$ is the proposed magnitude of additive noise, the testing is similar to that used in algorithm $\mathcal{S}$. To be concrete, for a particular discretization $B^{(j)}$, we can find the bin containing the data median $m(D)$, denoted by $B^{(j)}(D)$. Consider the following query:

$\mathbf{Q_1}$: How many data points need to change in order to get a new database $\hat{D}$ such that $m(\hat{D}) \notin B^{(j)}(D)$?

Clearly, the sensitivity of query Q_1 is at most 1, for the same reason as in the proof of Lemma 8. Let $A_1^{(j)}(D)$ be the answer to Q_1, under discretization $B^{(j)}$, let $R_1^{(j)}(D) = A_1^{(j)}(D) + z_1$, where $z_1 \sim \text{Lap}(1/\varepsilon)$. If $A_1 \leq 2$, then adding noise calibrated to h will ensure differential privacy because $|m(D') - m(D)| \leq h$ for all adjacent D'. The algorithm proceeds with noise calibrated to h if R_1 is large; otherwise it returns $\perp$ or looks for another discretization.

The algorithm $\mathcal{M}$

- **Input:** (D, n, ε, s).
- **Algorithm**
  1. If $s = \Lambda$ then set $s = \mathcal{S}(D, n, \varepsilon)$.
  2. If $s = \perp$, then return $\perp$, otherwise let $h = sn^{-1/3}$ as defined earlier. [a]
  3. For each discretization, $(j = 1, 2)$
     - Find $B^{(j)}(D)$.
     - Compute $R_1^{(j)}$.
     - If $R_1^{(j)} \leq \ln^2 n + 1$, let $m^{(j)} = \perp$. Otherwise let $m^{(j)} = m(D) + z_m^{(j)}$, where $z_m^{(j)}$ is randomly drawn from $\text{Lap}(h/\varepsilon)$.
  4. If $m^{(1)} \neq \perp$, return $m^{(1)}$.
     Otherwise return $m^{(2)}$.

[a] If $s = 0$, we can define $h(0)$ to be any positive number, e.g., can choose $h(0) = n^{-\frac{1}{2}}$. Since h is a fixed mapping and s is (ε, δ)-differentially private, we have h is also (ε, δ)-differentially private.

As with Algorithm $\mathcal{S}$, this algorithm can be optimized by only computing $m^{(2)}$ if $m^{(1)} = \perp$.

Theorem 12. *(a) The algorithm $\mathcal{M}$ is $(6\varepsilon, \nu(n))$-differentially private.*

(b) The computation cost for $\mathcal{M}$ is $O(n)$ assuming the data are sorted.

(c) Under the conditions in theorem 7 (c), and if F is differentiable with positive derivative at the median, then

$$\tilde{P}(\mathcal{M}(D) = \perp) = O(n^{-\varepsilon \ln n}),$$

and

$$\mathcal{M}(D) \xrightarrow{\tilde{P}} m(F), \quad \text{as } n \to \infty.$$

Apparently, everything is analogous to the algorithm $\mathcal{S}$, except that in Step 1 we call $\mathcal{S}$ to get s in order to obtain a reasonable scale of precision at which the median will be released.

Proof of theorem 12 We first study the query Q_1. Q_1 involves s, the output of $\mathcal{S}$, therefore the statement about Q_1 should be conditional on s.

Lemma 13. *The sensitivity of Q_1 is at most 1, given s.*

Proof. The proof is the same as that of Lemma 8. □

For a given $s \neq 0$, if $A_1(D) \geq 2$, then changing 1 data point can change the median by at most $h = s/n^{1/3}$. Here we define a partition of the possible values of h, i.e., $\mathbf{R}^{\geq 0}$, into two sets, based on a given database D:

Definition 14. $\mathcal{H}_D^{(j)} = \left\{ s \in \mathbf{R}^{\geq 0} : A_1^{(j)}(D) \geq 2 \right\}$.

In the following discussion we use $\mathcal{H}^{(j)} = \mathcal{H}_D^{(j)}$ for short.

We have the following simple lemma analogous to lemma 10 (b):

Lemma 15. *For any $s \in \mathbf{R}^{\geq 0} \backslash \mathcal{H}_D^{(j)}$, we have $P\left(m^{(j)} \neq \perp | s\right) \leq \frac{1}{2} n^{-\varepsilon \ln n}$.*

Proof. Same as in Lemma 10 (b). □

Lemma 16. *For $j = 1, 2$, the computation of $m^{(j)}$ is $(5\varepsilon, n^{-\varepsilon \ln n})$-differentially private.*

Proof. Since it does not matter which disretization to look at, we omit the index j and use $\tilde{m}$ in the place of $m^{(j)}$. All the indices j in the following proof refer to the discretization in the Algorithm $\mathcal{S}$. The notation "$\mathcal{S} = \perp$" is shorthand for "the output of Algorithm $\mathcal{S}$ is $\perp$."

Starting from the $\perp$ case, by Lemma 10 (a)

$$\begin{aligned}
& P(\tilde{m} = \perp) \\
=& P(\mathcal{S} = \perp) + P(\mathcal{S} \neq \perp, \tilde{m} = \perp) \\
\leq& e^{\varepsilon} P'(\mathcal{S} = \perp) + P(\mathcal{S} \neq \perp, \tilde{m} = \perp) \\
=& e^{\varepsilon} P'(\mathcal{S} = \perp) + P\left(s^{(1)} \neq \perp, \tilde{m} = \perp\right) \\
& + P\left(s^{(1)} = \perp, s^{(2)} \neq \perp, \tilde{m} = \perp\right).
\end{aligned} \tag{9}$$

For $j = 1, 2$, define $\mathcal{D}_0^{(j)} = \{D : A_0^{(j)}(D) \leq 2\}$. When $D \notin \mathcal{D}_0^{(1)}$, we have, by Lemma 10 (b),

$$\begin{aligned}
& P\left(s^{(1)} \neq \perp, \tilde{m} = \perp\right) \\
\leq& P(s^{(1)} \neq \perp) \\
\leq& \frac{1}{2} n^{-\varepsilon \ln n}.
\end{aligned} \tag{10}$$

When $D \in \mathcal{D}_0^{(1)}$, we have, by Corollary 11 and Lemma 13,

$$\begin{aligned}
& P\left(s^{(1)} \neq \perp, \tilde{m} = \perp\right) \\
=& P\left(s^{(1)} \neq \perp, R_1 \leq \ln^2 n + 1\right) \\
=& \int_{s>0} P\left(R_1 \leq \ln^2 n + 1\right) p\left(s^{(1)} = s\right) \\
\leq& \int_{s>0} e^{\varepsilon} P'\left(R_1 \leq \ln^2 n + 1\right) e^{2\varepsilon} p'\left(s^{(1)} = s\right) \\
=& e^{3\varepsilon} P'\left(s^{(1)} \neq \perp, R_1 \leq \ln^2 n + 1\right) \\
=& e^{3\varepsilon} P'\left(s^{(1)} \neq \perp, \tilde{m} = \perp\right).
\end{aligned} \tag{11}$$

Combining (10) and (11), we have

$$\begin{aligned}
& P(s^{(1)} \neq \perp, \tilde{m} = \perp) \\
\leq& e^{3\varepsilon} P'(s^{(1)} \neq \perp, \tilde{m} = \perp) + \frac{1}{2} n^{-\varepsilon \ln n}.
\end{aligned} \tag{12}$$

As for the third term on the right hand side of (9), apply exactly the same argument as in (10), (11), and (12) to obtain

$$\begin{aligned}&P\left(s^{(2)}\neq\perp,\tilde{m}=\perp\middle|\, s^{(1)}=\perp\right)\\ \leq& e^{3\varepsilon}P'\left(s^{(2)}\neq\perp,\tilde{m}=\perp\middle|\, s^{(1)}=\perp\right)+\frac{1}{2}n^{\varepsilon\ln n}.\end{aligned}\tag{13}$$

Finally, we have

$$\begin{aligned}&P\left(s^{(1)}=\perp,s^{(2)}\neq\perp,\tilde{m}=\perp\right)\\ =&P\left(s^{(1)}=\perp\right)P\left(s^{(2)}\neq\perp,\tilde{m}=\perp\middle|\, s^{(1)}=\perp\right)\\ \leq& P\left(s^{(1)}=\perp\right)\times\left(e^{3\varepsilon}P'\left(s^{(2)}\neq\perp,\tilde{m}=\perp\middle|\, s^{(1)}=\perp\right)+\frac{1}{2}n^{-\varepsilon\ln n}\right)\\ \leq& e^{\varepsilon}P'\left(s^{(1)}=\perp\right)\times e^{3\varepsilon}P'\left(s^{(2)}\neq\perp,\tilde{m}=\perp\middle|\, s^{(1)}=\perp\right)+\frac{1}{2}n^{-\varepsilon\ln n}\\ \leq& e^{4\varepsilon}P'\left(s^{(1)}=\perp,s^{(2)}\neq\perp,\tilde{m}=\perp\right)+\frac{1}{2}n^{-\varepsilon\ln n}.\end{aligned}\tag{14}$$

So putting (12) and (14) on the right hand side of (9) we have

$$P(\tilde{m}=\perp)\leq e^{4\varepsilon}P'(\tilde{m}=\perp)+n^{-\varepsilon\ln n}.\tag{15}$$

Next we consider non-$\perp$ case, i.e., for some $C\subseteq\mathbf{R}$ consider $P(\tilde{m}\in C)$. Now we have

$$\begin{aligned}&P\left(\tilde{m}\in C\right)\\ =&P\left(\mathcal{S}\neq\perp,\tilde{m}\in C\right)\\ =&P\left(s^{(1)}\neq\perp,\tilde{m}\in C\right)+P\left(s^{(1)}=\perp,s^{(2)}\neq\perp,\tilde{m}\in C\right).\end{aligned}\tag{16}$$

Let us investigate the first term on the right hand side of (16).

- If $D_1\notin\mathcal{D}_0^{(1)}$, we have, by Lemma 10 (b)

$$P\left(s^{(1)}\neq\perp,\tilde{m}\in C\right)\leq P\left(s^{(1)}\neq\perp\right)\leq\frac{1}{2}n^{-\varepsilon\ln n}.\tag{17}$$

- On the other hand, if $D_1\in\mathcal{D}_0^{(1)}$, then, by Corollary 11 and Lemmas 13 and 15,

$$\begin{aligned}&P\left(s^{(1)}\neq\perp,\tilde{m}\in C\right)\\ =&\int p\left(s^{(1)}=s\right)P\left(R_1>\ln^2 n+1\right)P\left(z_m^{(1)}+m\in C\right)ds\\ =&\int_{s\in\mathcal{H}_D^{(1)}}p\left(s^{(1)}=s\right)P\left(R_1>\ln^2 n+1\right)P\left(z_m^{(1)}+m\in C\right)ds\\ &+\int_{s\notin\mathcal{H}_D^{(1)}}p\left(s^{(1)}=s\right)P\left(R_1>\ln^2 n+1\right)P\left(z_m^{(1)}+m\in C\right)ds\\ \leq&\int_{s\in\mathcal{H}_D^{(1)}}e^{2\varepsilon}p'\left(s^{(1)}=s\right)e^{\varepsilon}P'\left(R_1>\ln^2 n+1\right)e^{\varepsilon}P'\left(z_m^{(1)}+m\in C\right)ds\end{aligned}$$

$$
\begin{aligned}
&+\int_{s\notin\mathcal{H}_D^{(1)}} p\left(s^{(1)}=s\right)\frac{1}{2}n^{-\varepsilon\ln n}P\left(z_m^{(1)}+m\in C\right)ds\\
\leq& e^{4\varepsilon}P'\left(s^{(1)}\neq\perp,\tilde{m}\in C\right)+\frac{1}{2}n^{-\varepsilon\ln n}. \qquad (18)
\end{aligned}
$$

Combining (17) and (18), we have

$$
P\left(s^{(1)}\neq\perp,\tilde{m}\in C\right)\leq e^{4\varepsilon}P'\left(s^{(1)}\neq\perp,\tilde{m}\in C\right)+\frac{1}{2}n^{-\varepsilon\ln n}. \qquad (19)
$$

The same analysis also gives an inequality concerning the second term in the right hand side of (16),

$$
\begin{aligned}
&P\left(s^{(2)}\neq\perp,\tilde{m}\in C\middle|\, s^{(1)}=\perp\right)\\
\leq& e^{4\varepsilon}P'\left(s^{(2)}\neq\perp,\tilde{m}\in C\middle|\, s^{(1)}=\perp\right)+\frac{1}{2}n^{-\varepsilon\ln n},
\end{aligned}
$$

indicating that

$$
\begin{aligned}
&P\left(s^{(1)}=\perp,s^{(2)}\neq\perp,\tilde{m}\in C\right)\\
=&P\left(s^{(1)}=\perp\right)P\left(s^{(2)}\neq\perp,\tilde{m}\in C\middle|\, s^{(1)}=\perp\right)\\
\leq&P\left(s^{(1)}=\perp\right)\times\left(e^{4\varepsilon}P'\left(s^{(2)}\neq\perp,\tilde{m}\in C\middle|\, s^{(1)}=\perp\right)+\frac{1}{2}n^{-\varepsilon\ln n}\right)\\
\leq&e^{\varepsilon}P'\left(s^{(1)}=\perp\right)\times e^{4\varepsilon}P'\left(s^{(2)}\neq\perp,\tilde{m}\in C\middle|\, s^{(1)}=\perp\right)+\frac{1}{2}n^{-\varepsilon\ln n}\\
\leq&e^{5\varepsilon}P'\left(s^{(1)}=\perp,s^{(2)}\neq\perp,\tilde{m}\in C\right)+\frac{1}{2}n^{-\varepsilon\ln n}. \qquad (20)
\end{aligned}
$$

Put (19) and (20) in the right hand side of (16), we have

$$
P\left(\tilde{m}\in C\right)\leq e^{5\varepsilon}P'\left(\tilde{m}\in C\right)+n^{-\varepsilon\ln n}. \qquad (21)
$$

□

Proof of theorem 12. (a) The result is straightforward based on Lemma 16 and the fact that $m^{(1)}$ and $m^{(2)}$ are independent, using similar argument as in the proof of Theorem 7 part (a). A detailed proof under a general framework is given in Section 8.

(b) Suppose the data points in D are sorted as $x_{(1)}\leq x_{(2)}\leq ....$ To compute $A_1^{(j)}(D)$, it is sufficient to compute $\min\{i: x_{(i)}\in B_{k_j}^{(j)}\}$ and $\max\{i: x_{(i)}\in B_{k_j}^{(j)}\}$, which can be done in $O(n)$ time.

(c) The proof is essentially the same as that of Theorem 7 (c), except doing the conditional analysis on the event

$$
E_0: \left\{\mathcal{S}(D)\in\left[\frac{1}{2}n^{-1/20}IQR(F),2n^{1/20}IQR(F)\right]\right\}.
$$

By the results in Theorem 7 (c) and (d),

$$\tilde{P}(E_0) \geq 1 - O(n^{-c_1 \ln n}).$$

Let $l = m(F) - n^{-2/5}$, $r = m(F) + n^{-2/5}$, $l' = m(F) - 2n^{-2/5}$, $r' = m(F) + 2n^{-2/5}$. By differentiability of F, we can find constant $\xi > 0$ such that for all large enough n,

1. $F'(x) \geq \xi$ on (l', r').
2. $\xi nh/4 > 4\ln^2 n + 4$.

Define similar random events as in theorem 7 (c):

(i) $E_1 = \{m(D) \in [l, r]\}$.

$$\begin{aligned} P_F(E_1) =& 1 - P_F\left(m(D) < l, \text{ or } m(D) > r\right) \\ \geq& 1 - P_F(m(D) < l) - P_F(m(D) > r) \\ \geq& 1 - 2P_F\left(\sup_x |F(x) - F_n(x)| > \xi n^{-2/5}\right) \\ \geq& 1 - c_0 e^{-c_1 n^{1/5}}. \end{aligned} \tag{22}$$

(ii) $E_2 = \left\{\min\left(|D \cap (l', l)|, |D \cap (r, r')|\right) \geq \xi n^{3/5}/2\right\}$, using Hoeffding's inequality as we did in the proof of Theorem 7 (c), part (ii), we have

$$P_F(E_2) \geq 1 - c_0 e^{-c_1 n^{1/5}}.$$

Note that on E_0, $h = \Omega\left(n^{-23/60}\right)$, then for large enough n, $n^{-2/5} < \frac{1}{8}h$. By our construction of $B^{(j)}$, $j = 1, 2$, there exists at least one $B^{(j)}$, such that $[l', r'] \subset B^{(j)}(D)$. Then on $E_0 \cap E_1 \cap E_2$ we have $A_1^{(j)}(D) \geq \xi n^{3/5}/2 \geq 2\ln^2 n + 2$. So we have

$$P\left(m^{(j)}(D) = \perp | E_0 \cap E_1 \cap E_2\right) \leq O\left(n^{-\varepsilon \ln n}\right).$$

Remember that $\tilde{P}(E_0 \cap E_1 \cap E_2) \geq 1 - O(n^{-c_1 \ln n})$, we have

$$\tilde{P}\left(m^{(j)}(D) = \perp\right) \leq O\left(n^{-c_1 \ln n}\right),$$

which indicates

$$\tilde{P}\left(\mathcal{M}(D) = \perp | \mathcal{S}(D) \neq \perp\right) \leq O\left(n^{-c_1 \ln n}\right).$$

On the other hand, we know that under the assumptions, $\tilde{P}\left(\mathcal{S}(D)\right) \neq \perp \geq 1 - O\left(n^{-c_1 \ln n}\right)$, then we finally have the desired result in part (c) of Theorem 12

$$\tilde{P}\left(\mathcal{M}(D) = \perp\right) \leq O\left(n^{-c_1 \ln n}\right).$$

The other claim holds trivially based on the above results.

□

7 Linear regression.

The linear regression model is

$$Y = X^T\beta + \phi,$$

where $Y \in \mathbf{R}^1$, $X, \beta \in \mathbf{R}^p$, $P(||X|| > 0) = 1$, and $\phi \in \mathbf{R}^1$ is independent of X and its distribution is continuous and symmetric about 0.

The data set $D = \{(x_i, y_i)_{i=1}^n\}$ consists of n iid samples from the joint distribution of (X, Y), and the inference task is to estimate β. In this section we first introduce a simple short-cut regression algorithm which fully utilizes the previous scale and median algorithms, then we describe a differentially private algorithm based on a particular robust regression estimator.

7.1 A short-cut method

A simple way to carry out an (ε, δ)-differentially private regression analysis is to transform the regression problem to a simpler one which has been worked out already, such as the scale and/or median. We propose the following "short-cut" regression algorithm.

At a high level, the algorithm first randomly partitions the inputs into disjoint blocks of p data points each. An approximation to β is computed from each block. This gives n/p independent approximations to β. For each coordinate, run Algorithm $\mathcal{M}$ on a dataset consisting of the n/p different values obtained for this coordinate to obtain a single output value for this coordinate. Output the vector consisting of these p coordinates.

We now define $\tilde{\phi}$, which, as we will presently explain, gives an approximation to β. Intuitively, the "p copies of X" mentioned below correspond to the data items in one of the n/p blocks in the informal description above (recall each data item is of the form (x_i, y_i) for $x_i \in \mathbf{R}^p$). Let $\tilde{\phi} = \mathbf{X}^{-1}\vec{\phi}$, where $\vec{\phi} = \{\phi_1, \ldots, \phi_p\}^T$ is a $p \times 1$ vector consisting of p i.i.d. copies of ϕ, and $\mathbf{X} = (X_1, \ldots, X_p)^T$ is a matrix that consists of p i.i.d copies of X (assume that $\mathbf{X}$ is invertible with probability 1, which is just requiring the design matrix is of full rank with probability 1). Also letting $\mathbf{Y} = \mathbf{X}\beta + \vec{\phi}$ be the associated vector of Y, then $\mathbf{X}^{-1}\mathbf{Y}$ is an approximation to β. In fact, multiply both sides of the equation

$$\mathbf{Y} = \mathbf{X}\beta + \vec{\phi}$$

by $\mathbf{X}^{-1}$ to get $\mathbf{X}^{-1}\mathbf{Y} = \beta + \mathbf{X}^{-1}\vec{\phi}$. By the assumption that ϕ is symmetrically distributed and independent of X, we have that each coordinate of $\tilde{\phi}$ is symmetrically distributed. For $1 \le d \le p$, let $\mathrm{IQR}_d = \mathrm{IQR}(\tilde{\phi}_d)$. We will use this in the next section.

For now, we obtain the following short-cut linear regression algorithm:

The algorithm $\mathcal{R}_S$

- **Input:** (D, n, ε).
- **Algorithm**
  1. Randomly partition the data points into $m = \lfloor n/p \rfloor$ groups of size p.
  2. In each group, compute the least square estimator of β, i.e., $\mathbf{X}^{-1}\mathbf{Y}$, and store it.
  3. Let $D^{(d)}, 1 \leq d \leq p$ be the dataset consisting of all the dth coordinates of the stored vectors.
  4. For $1 \leq d \leq p$

     Run $\mathcal{M}$ on $\left(D^{(d)}, \lfloor n/p \rfloor, \varepsilon, \Lambda\right)$. Return $\mathcal{M}(D^{(d)})$ as the estimate for β_d, where $\beta = (\beta_1, \ldots, \beta_p)^T$.

Readers familiar with the Subsample-and-Aggregate framework of Nissim, Raskhodnikova, and Smith [17] can view Algorithm $\mathcal{R}_S$ in this light: the observations are partitioned into blocks (subsampling), where each block yields a value for β; these values are then aggregated by p invocations of $\mathcal{M}$.

Algorithm $\mathcal{R}_S$ is a direct application of the algorithm $\mathcal{M}$, and based on our previous analysis, $\mathcal{R}_S$ is $(6p\varepsilon, \nu(n))$-differentially private (say, for constant p, independent of n). Clearly the computation time is $O(n)$, assuming the data are sorted, $O(n \log n)$ otherwise, and under mild distributional assumptions on X and ϕ, such as continuous and positive density, the probability of getting $\perp$ is negligible in n and the estimator is consistent. The proof of the differential privacy of $\mathcal{R}_S$ is a bit different from the previous ones and we will prove a more general form in section 9.

7.2 A Robust Regression Estimator

In this subsection we follow the framework: that is, starting from a robust regression estimator of β, proposing a scale of additive noise based on the order of magnitude of an expected deviation, and then testing if that scale is enough for differential privacy. The particular robust regression estimator $\mathbf{H}$ whose output $\hat{\beta}$ is defined as follows (see [12] for more detail):

$$\hat{\beta} = \arg\min_{\beta} f_D(\beta), \quad \text{where } f_D(\beta) = \sum_{i=1}^{n} |y_i - x_i^T \beta| / ||x_i||. \tag{23}$$

The specific output $\hat{\beta}$ may depend on the optimization algorithm used.

Here for presentation simplicity we illustrate the algorithm and discuss its validity in the case $p = 2$, and the generalization to other value of p is straightforward.

Suppose we are given any algorithm which computes $\beta(D) \in B(D)$, where $B(D)$ is the whole solution set to the optimization problem (23). Now $\beta \in \mathbf{R}^2$, and a discretization in $\mathbf{R}^2$ should be the product of two discretizations in $\mathbf{R}^1$ which corresponds to the two coordinates of β. As a result, $\mathbf{R}^2$ is discretized into rectangle cells:

$$\{C_{kl} = [kh_1, (k+1)h_1) \times [lh_2, (l+1)h_2)\}_{k,l \in \mathbf{Z}},$$

where $h_d,\ d = 1, 2$ is the proposed magnitude of additive noise for each coordinate of β. Similarly, in order to avoid the "end point problem" (i.e., the situation that $\beta(D)$ happens to be on the

edge of the cell), one can consider multiple discretization. Since for each coordinate two different discretizations $B_k^{(1)} = [kh, (k+1)h)$ and $B_k^{(2)} = [(k-0.5)h, (k+0.5)h)$ would be sufficient, we will need to consider four product discretizations (in general p-dimensional problem this number is 2^p). For definiteness we list the four product discretizations as follows:

$$
\begin{aligned}
C^{(1)} &= \left\{ C_{kl}^{(1)} = [kh_1, (k+1)h_1) \times [lh_2, (l+1)h_2) \right\}_{k,l \in \mathbf{Z}}, \\
C^{(2)} &= \left\{ C_{kl}^{(2)} = [(k-0.5)h_1, (k+0.5)h_1) \times [lh_2, (l+1)h_2) \right\}_{k,l \in \mathbf{Z}}, \\
C^{(3)} &= \left\{ C_{kl}^{(3)} = [kh_1, (k+1)h_1) \times [(l-0.5)h_2, (l+0.5)h_2) \right\}_{k,l \in \mathbf{Z}}, \\
C^{(4)} &= \left\{ C_{kl}^{(4)} = [(k-0.5)h_1, (k+0.5)h_1) \times [(l-0.5)h_2, (l+0.5)h_2) \right\}_{k,l \in \mathbf{Z}}.
\end{aligned}
$$

Now for each discretization define $C^{(j)}(D)$ to be the bin in the jth discretization such that $\beta(D) \in C^{(j)}(D)$. It is generally hard to track $\beta(D)$ since it depends on which particular optimization algorithm is used, but it is easier to consider $B(D)$ which is an intrinsic property of the optimization problem determined totally by D. As we will see, in the cases of interest to us $B(D)$ will typically be small and covered by one of $\{C^{(j)}(D)\}_{j=1,\ldots,4}$. Here the testing query is slightly different from the previous ones:

$\mathbf{Q_2}$: How many data points need to add or delete in order to get a database $\hat{D}$ such that $B(\hat{D})$ is not covered by $C^{(j)}(D)$?

Note that we view (h_1, h_2) as fixed, so we don't explicitly list them as inputs to $\mathbf{Q_2}$. $A_2^{(j)}$ (the true answer) and $R_2^{(j)}$ (the true answer plus noise) are defined similarly as before. Because changing one data point could be viewed as equivalent to deleting one original data point and adding one with the modified value, if $A_2^{(j)} \geq 3$ for some j, then for all adjacent databases D' such that $|D'| = |D|$, we have $|\beta_d(D') - \beta_d(D)| \leq h_d$ for $d = 1, 2$.

Then a differentially private algorithm for this estimator can be described as:

The algorithm $\mathcal{R}_H$

- **Input:** (D, n, ε).
- **Algorithm:**

1. Compute $\beta(D)$.

2-4. The same as steps 1-3 in $\mathcal{R}_S$ (Partition; within each group compute β; define the sets $D^{(d)}, d = 1, 2.$).

5. For $1 \le d \le 2$

   Run $\mathcal{S}$ on $\left(D^{(d)}, \lfloor n/2 \rfloor, \varepsilon\right)$. If any of the outputs is $\perp$, then return $\perp$. Otherwise denote the outputs as s_d, $1 \le d \le 2$, and let $h_d = s_d/n^{1/4}$. Also, if $s_d = 0$, let $h_d = n^{-1/2}$.

6. For $1 \le j \le 4$
   - Compute $R_2^{(j)}(D, (h_1, h_2))$.
   - If $R_2^{(j)}(D, (h_1, h_2)) \le \ln n^2 + 2$, let $\beta^{(j)} = \perp$. Otherwise $\beta^{(j)} = \beta(D) + z^{(j)}$, where $z^{(j)} = (z_1^{(j)}, z_2^{(j)})$ and $z_d^{(j)} \sim \text{Lap}(h_d/\varepsilon)$ for $d = 1, 2$.

7. Find the smallest j such that $\beta^{(j)} \neq \perp$. If such a j exists, return $\mathcal{R}_H(D) = \beta^{(j)}$, otherwise return $\mathcal{R}_H(D) = \perp$.

We have the following theorem ensuring the property of algorithm $\mathcal{R}_H$:

Theorem 17. *(a) The algorithm $\mathcal{R}_H$ is $(11\varepsilon, \nu(n))$-differentially private*[13].

(b) $\mathcal{R}_H$ runs in $O(n^3)$ time[14].

(c) If

(i) For all $1 \le d \le p$, $\tilde{\phi}_d$ has continuous and positive density; and

(ii) $f(\beta) = E_F|Y - X^T\beta|/||X||$ is twice continuously differentiable, and $E_F XX^T/||X||$ is positive definite, then

$$\tilde{P}(\mathcal{R}_H(D) = \perp) = O(n^{-c \ln n}).$$

and

$$\mathcal{R}_H(D) \xrightarrow{\tilde{P}} \beta^*,$$

where β^* is the true value of regression coefficient in the model.

Remark 18. In the algorithm $\mathcal{R}_H$, the magnitude of noise does not have to be on the order of $n^{-1/4}$ (for general p our conservative choice is $n^{-1/2p}$, although $n^{-1/2+c}$ would work). As can be seen later in the proof, under the assumptions of Theorem 17, one can choose $h_d = s_d n^{-1/2+\zeta}$, for some small positive constant ζ. The value $n^{-1/4}$ is chosen from a practical perspective that the discretized cell contains approximately $1/\sqrt{n}$ proportion of the data (the group of β's generated by the random partition of D).

[13] The number 11 becomes $2^p + 3p + 1$ for general value of p.

[14] For general value of p, the running time is $O(n^{p+1})$

Proof of Theorem 17. The proof of part (a) (differential privacy) is analogous to that of theorem 12 (a), and we omit it here, the only possible difference is that the analysis is conditional on the random partition. We give a general proof in section 9.

To show part (b) (ease of computation), we need to study the answer to query $\mathbf{Q}_2$. Let us start from another query

> Q_β : For a particular $\beta \in \mathbf{R}^2$, how many data points need to be added or deleted in order to get a database $\hat{D}$, such that $\beta \in B(\hat{D})$?

Let $A(D)$ be the answer to Q_2, on database D. Clearly $A(D) = \inf_{\beta \notin C(D)} A_\beta(D)$, where $A_\beta(D)$ is the answer to the query Q_β on database D.

To compute $A_\beta(D)$, we need to explore the structure of function $f_D(\beta)$. A first observation is that $f_D(\beta)$ is convex and piecewise linear. The convexity is trivial since each term in the sum in (23) is convex. To see piecewise linearity, define $\ell_i = \{\gamma \in \mathbf{R}^2 : y_i - x_i^T \gamma = 0\}$. Then the ith term in (23) is just the distance from β to ℓ_i. The space $\mathbf{R}^2$ is partitioned into $O(n^2)$ convex regions by these n lines, and the lines themselves are cut by each other into $O(n^2)$ line segments or half lines with $O(n^2)$ intersections. In the later discussion, the terms "region", "line segment", "half line" and "intersection" all refer to those described above.

Note inside each region, $f_D(\beta)$ is a linear function, since now for each i, $|y_i - x_i^T \beta|/||x_i||$ is linear confined in the region.

The minimum of a convex function can be characterized by the *subgradient*:

Definition 19 (Subgradient)**.** A vector γ is called a subgradient of f at β, if

$$f(\beta + \Delta) \geq f(\beta) + \gamma^T \Delta, \quad \forall \Delta.$$

The set of all subgradients at β is called *subdifferential*, denoted by $\partial f(\beta)$. Clearly, if f is convex and differentiable at β, then $\partial f(\beta) = \{df/d\beta\}$, a set with a single element.

A result in convex optimization gives the characterization of the minimum of f in terms of ∂f:

Theorem 20 ([4])**.** *For any convex function f and $\beta \in$ Domain(f), β is a global minimizer of f if and only if $0 \in \partial f(\beta)$.*

The next question is how to compute $\partial f_D(\beta)$. For β in the interior of a region, $f_D(\beta)$ is linear inside that region, so $\partial f(\beta)$ has only a single vector $df/d\beta$. When β is on the line segments, half lines or intersections, $f_D(\beta)$ is not differentiable and $\partial f_D(\beta)$ contains multiple elements. Now such a β is surrounded by several regions. For example, if β is in the interior of a line segment or half line, it is surrounded by two neighboring regions; if β is the intersection of two lines, there are four surrounding regions. Denote these surrounding regions by $R_r, r = 1, \ldots, r_0$, and let $f_r(\cdot)$ be the linear function that agrees with $f_D(\cdot)$ on R_r. Then we have, on a small open neighborhood of β,

$$f_D = \max \{f_r, r = 1, \ldots, r_0\},$$

and

$$f_r(\beta) = f_D(\beta), \quad r = 1, \ldots, r_0.$$

Another basic result in convex analysis and optimization gives the description of $\partial f(\beta)$:

Theorem 21. *If $f = \max_{r=1,\ldots,r_0} f_r$, then*

$$\partial f(\beta) = \mathbf{CH} \bigcup_{r: f_r(\beta) = f(\beta)} \{\partial f_r(\beta)\},$$

where **CH** *means the convex hull.*

Therefore, for any $\beta \in \bigcup_{i=1}^{n} \ell_i$, i.e., those not in the interior of a region, denote the derivatives of the linear functions f_r by γ_r for $r = 1, \ldots, r_0$, then $\partial f_D(\beta) = \mathbf{CH}\{\gamma_1, \ldots, \gamma_{r_0}\}$.

With these preliminary knowledge about the structure of f_D and its minimum, we have the following lemma:

Lemma 22. $A_\beta(D) = \left\lceil \inf_{\gamma \in \partial f_D(\beta)} ||\gamma|| \right\rceil$.

Proof. Suppose β is surrounded by r_0 regions[15], namely, $R_r,\ r = 1, \ldots, r_0$. For $r = 1, \ldots, r_0$, let γ_r be the derivative of f_r at β, where f_r is the linear function that agrees with f_D on R_r.Then $\partial f_D(\beta) = \mathbf{CH}(\gamma_1, ..., \gamma_r)$, where $\mathbf{CH}(\cdot)$ denotes the convex hull.

Let $\hat{D}$ be another database obtained by changing k data points of D, and the corresponding objective function in (23) is $f_{\hat{D}}(\beta)$. Note that adding/deleting data points is equivalent to adding/deleting the lines ℓ_i, since each data point (x_i, y_i) corresponds to a line ℓ_i. A consequence of such a adding/deleting is that two regions might merge (in case of deleting), and a region might be cut into smaller regions (in case of adding). Note also that modifying a data point is equivalent to deleting it and inserting a new point with the modified value. As a result, the set of regions that surrounds β, namely $\{\mathbf{R}_r, r = 1, \ldots, r_0\}$, might be changed to $\{\hat{R}_t, t = 1, \ldots, t_0\}$. Let g_t be the linear function that agrees with $f_{\hat{D}}$ on $\hat{R}_t$, and λ_t be its derivative at β. We have

$$\partial f_{\hat{D}}(\beta) = \mathbf{CH}(\lambda_1, ..., \lambda_{t_0}),$$

and

$$\lambda_t = \gamma_{r_t} + \eta_t, \quad t = 1, ..., t_0, \quad r_t \in \{1, ..., r_0\},$$

where $\eta_t = \frac{\partial g_t(\beta)}{\partial \beta} - \frac{\partial f_{r_t}(\beta)}{\partial \beta}$ is the change of ∂f_D (also ∇f_D) at some particular $\beta_t \in R_{r_t} \cap \hat{R}_t$ incurred by changing the data points.

Note that if β minimizes $f_{\hat{D}}(\cdot)$, then $0 \in \partial f_{\hat{D}}(\beta)$, that is, there exists $\mu_1, ..., \mu_{t_0} \geq 0$, $\sum_{t=1}^{t_0} \mu_t = 1$, such that

$$0 = \sum_{t=1}^{t_0} \mu_t \lambda_t.$$

Then we have

$$\begin{aligned}
& 0 = \sum_{t=1}^{t_0} \mu_t (\gamma_{r_t} + \eta_t) \\
\Rightarrow & \sum_{t=1}^{t_0} \mu_t \gamma_{r_t} = -\sum_{t=1}^{t_0} \mu_t \eta_t \\
\Rightarrow & \sum_{r=1}^{r_0} \mu'_r \gamma_r = -\sum_{t=1}^{t_0} \mu_t \eta_t, \qquad \mu'_r \geq 0, \quad \sum_r \mu'_r = 1 \\
\Rightarrow & \left|\left| \sum_{r=1}^{r_0} \mu'_r \gamma_r \right|\right| = \left|\left| \sum_{t=1}^{t_0} \mu_t \eta_t \right|\right| \leq \max_t ||\eta_t|| \\
\Rightarrow & \max_t ||\eta_t|| \geq \inf_{\gamma \in \partial f(\beta)} ||\gamma||.
\end{aligned}$$

[15] In case that β is in the interior of a region, $r_0 = 1$

The second implication follows by adding coefficients of the r_i, and the third because for all i, the ith term is bounded by $\mu_i \max_t ||\eta_t||$. Note that η_t is the change on the subgradient at a certain point $\beta_t \in R_{r_t} \cap \hat{R}_t$ incurred by changing k data points in D, and that the magnitude of change on the subgradient by a single addition or deletion is at most 1. So one needs to add or delete at least $\lceil ||\eta_t|| \rceil$ data points to induce a change on the subgradient by η_t. As a result, to make β to be the solution of (23), one needs to change at least $\left\lceil \inf_{\gamma \in \partial f_D(\beta)} ||\gamma|| \right\rceil$ data points. Thus $A_\beta(D) \geq \left\lceil \inf_{\gamma \in \partial f_D(\beta)} ||\gamma|| \right\rceil$.

On the other hand, let $\gamma_0 \in \partial f_D(\beta)$ such that $||\gamma_0|| = \inf_{\gamma \in \partial f_D(\beta)} ||\gamma||$. Then one can always add $\left\lceil \inf_{\gamma \in \partial f_D(\beta)} ||\gamma|| \right\rceil$ data points such that $\partial f_{\hat{D}}(\beta) = \partial f_D(\beta) - \gamma_0$, making β the solution of (23). To see this, let $k = \lceil ||\gamma|| \rceil$. One can always find (x_i', y_i'), $i = 1, ..., k$, such that $||x_i'|| = 1, \sum_{i=1}^k x_i' = \gamma$ and $y_i' > x_i'^T \beta$. Then $f_{\hat{D}}(\beta) = f_D(\beta) + \sum_{i=1}^k (y_i' - x_i'^T \beta)$. As a result

$$\partial f_{\hat{D}}(\beta) = \partial f_D(\beta) - \sum_{i=1}^k x_i = \partial f_D(\beta) - \gamma,$$

which implies $0 \in \partial f_{\hat{D}}(\beta)$. Therefore $A_\beta(D) \leq \left\lceil \inf_{\gamma \in \partial f_D(\beta)} ||\gamma|| \right\rceil$. □

As for part (b) of Theorem 17, a first observation is that to compute $A_2(D)$, it is enough to find the infimum of $A_\beta(D)$ among all $\beta \notin C^{(j)}(D)$, for any given j. Due to the piecewise linearity of f_D, $A_\beta(D)$ is the same for all β's inside a region, line segment, or half line. As a result, it is enough to consider all the regions, line segments, half lines and intersections. One further observation is that for any β inside a region R, one can always find a β' on the boundary of R, i.e., on the line segments, half lines or intersections. Then by the argument above we have $A_{\beta'}(D) \leq A_\beta(D)$, since $\partial f_D(\beta) \subseteq \partial f_D(\beta')$. So it suffices to focus on all the line segments and intersections. For each line, a pass through the data set can find out all the line segments, half lines and intersections on it, by simply computing its intersection with other lines. This takes $O(n)$ time in computing all the intersections, and another $O(n \log n)$ time to sort the intersections which gives the line segments and half lines on it. So it takes $O(n^2 \log n)$ time to compute and store all the line segments, half lines and intersections. For each of them, it takes $O(n)$ time to compute the subdifferential and hence $A_\beta(D)$. Finally, finding the smallest of them takes $O(n^2)$ time. Summing up, all of these can be done in $O(n^3)$ time ($O(n^{p+1})$for general p).

It remains to show part (c) of Theorem 17 (good behavior on nice distributions) . The main idea is the same as previous: use law of large numbers to ensure, with high probability, $A_\beta(D)$ are not far from its average over all random D drawn from F. However, here we have infinitely many β in the set $\mathbf{R}^2 \backslash C^{(j)}(D)$, which means we need to control the maximum of a stochastic process instead of finite number of random variables. To be concrete, one subgradient of f at β can be written as:

$$g(\beta) = \sum_{i=1}^n \text{sign}(y_i - x_i^T \beta) x_i / ||x_i||,$$

the norm of which, under the condition in part (c) of theorem 17, approximates $A_\beta(D)$ within a constant of 4, since there are at most two lines crossing β which indicates $\text{diam}(\partial f(\beta)) \leq 4$.

Then we need to show that $\min_{\beta \notin C^{(j)}(D)} ||g(\beta)|| \gg \ln^2 n$ for some j, with high probability. The scheme is similar to the proof of theorem 12 (c). Recall that

$$f(\beta) = E_F |Y - X^T \beta| / ||X||,$$

then our assumption ensures that f has a unique minimum β^*.

Consider the cell

$$L = \left[\beta_1^* - \frac{1}{8}h_1, \beta_1^* + \frac{1}{8}h_1\right] \times \left[\beta_2^* - \frac{1}{8}h_2, \beta_2^* + \frac{1}{8}h_2\right].$$

Define event $E_0 = \left\{s_d \in \left[\frac{1}{2}n^{-1/20}IQR_d, 2n^{1/20}IQR_d\right], 1 \le d \le p\right\}$. From the proof of Theorem 12 (c) we know that $\tilde{P}(E_0) \ge 1 - O(n^{-c_1 \ln n})$.

On E_0 we have $||h|| \to 0$, and by the strict convexity and differentiability of f, there exists constant c such that for large enough n,

$$||\nabla f(\beta)|| \ge c||h||, \quad \text{for all } \beta \in L^C. \tag{24}$$

Intuitively $g(\beta)$ (and hence $A_\beta(D)$) can be approximated by $||n\nabla f(\beta)||$, and L is small enough for large n, hence well covered by $C^{(j)}(D)$, for some j. So we expect that $||A_\beta(D)|| = \Omega(nh)$.

Rigorously, consider the stochastic process

$$g(D;\beta) = \frac{1}{n}\sum_{i=1}^{n} \text{sign}\left(\phi_i - X_i^T(\beta - \beta^*)\right) X_i/||X_i||.$$

Note that

$$ng(D;\beta) = \nabla f_D(\beta), \quad \text{for all } \beta \in \left(\bigcup_{i=1}^{n} \ell_i\right)^C,$$

as a result

$$A_\beta(D) \ge n||g(D;\beta)|| - 1, \quad \text{for all } \beta \in \left(\bigcup_{i=1}^{n} \ell_i\right)^C. \tag{25}$$

The sketch of proof is first to use the uniform bound on empirical processes to get a lower bound of $\inf_{\beta \in ((\bigcup \ell_i) \bigcup L)^C} A_\beta(D)$. Then we extend the result to all $\beta \in L^C$, using the continuity of the distribution of ϕ and X.

First let B be a countable dense subset of L^C, $\nabla_d f(\beta)$ and $g_d(D;\beta)$ be the dth coordinate of the corresponding vectors for $d = 1, 2$. The theory of empirical processes gives us the following lemma follows largely the argument used in [18, Ch II.3]:

Lemma 23. *For $d = 1, 2$*

$$P_F\left(\sup_{\beta \in B} |g_d(D;\beta) - \nabla_d f(\beta)| > n^{-1/3}\right) \le O\left(n^2 e^{-c_1 n^{1/3}}\right).$$

Denote by E_1 the event $\left\{\sup_{\beta \in B} |g_d(D;\beta) - \nabla_d f(\beta)| \le n^{-1/3}\right\}$. Then

$$P_F(E_1) \ge 1 - O\left(n^2 e^{-c_1 n^{1/3}}\right).$$

Next we confine the analysis on $E_0 \cap E_1$, where we have $||h|| = \Omega(n^{-3/10})$, so for large enough n, $n^{-1/3} \le \frac{1}{2}c||h||$. Then (24) and (25) implies

$$\inf_{\beta \in B} A_\beta(D) \ge n||\nabla f(\beta)|| - n||\nabla f(\beta) - g(D;\beta)|| - 1 \ge \frac{2 - \sqrt{2}}{2}cn||h|| - 1 = \Omega\left(n^{7/10}\right). \tag{26}$$

Because B is dense in L^C, (26) simply implies that the same bound hold for all $\beta \in ((\bigcup_{i=1}^n \ell_i) \bigcup L)^C$, i.e.,

$$\inf_{\beta \in ((\bigcup_{i=1}^n \ell_i) \bigcup L)^C} A_\beta(D) \geq \Omega\left(n^{7/10}\right).$$

Furthermore, for those $\beta \in \ell_i \cap L^C$ for some i, since there are at most two lines crossing β,[16] based on the fact that $\operatorname{diam}(\partial f_D(\beta)) \leq 4$ and that B is dense in L^C, we have

$$A_\beta(D) \geq \inf_{\beta \in ((\bigcup_{i=1}^n \ell_i) \bigcup L)^C} A_\beta(D) - 4.$$

Then we finally have

$$\inf_{\beta \in L^C} A_\beta(D) \geq \Omega\left(n^{7/10}\right). \tag{27}$$

Clearly (27) implies that $\beta(D) \in L$ because $A_{\beta(D)}(D) = 0$. Furthermore, by the construction of $C^{(j)}(D), j = 1, \ldots, 4$, and the definition of L, there exists at least one j, such that $L \subset C^{(j)}(D)$. Then (27) implies $A_2^{(j)}(D) \geq \Omega\left(n^{7/10}\right)$. So for large enough n, we have $A_2^{(j)}(D) \geq 2\ln^2 n + 2$, which implies $\tilde{P}(\mathcal{R}_H(D) = \perp | E_0 \cap E_1, s \neq \perp) = O(n^{-\varepsilon \ln n})$. But as shown in theorem 7 (c), $\tilde{P}(s \neq \perp) \geq 1 - O(n^{-c_1 \ln n})$ under our assumptions, as a result, we have the desired inequality for theorem 17 (c):

$$\tilde{P}(\mathcal{R}_H(D) = \perp) \leq O(n^{-c \ln n}).$$

8 (ε, δ)-PTR Functions

So far we have seen several examples of (ε, δ)-differentially private release for robust estimators. These mechanisms share the same spirit. First an insensitive magnitude is proposed, then we test privately whether this magnitude is big enough such that the an additive Laplacian random noise calibrated to it is enough to provide the ε-differential privacy. We call such a framework Propose-Test-Release (PTR). Now we study the in general random functions of the form $T(D, s)$, where D is a database and s is a second input used in computing a proposed bound on local sensitivity. This computation, call it $g(D, s)$, can be independent of D (as in Algorithm $\mathcal{S}$, where $g(D, s) = 1$ is the bin width used in the discretization of $\mathbf{R}$ for the logarithm of the interquartile range, or it can depend on D, as in Algorithm $\mathcal{M}$, where a scale $g(D, s) = \mathcal{S}(D)$ is computed and then used to obtain the bin width for testing the sensitivity of the median. From now on we always treat D and D' as non-random, and the probability is over the coin flips of the random function T, which are always independent of everything else. Similarly, $P(\cdot)$ refers to the probability when the input database is D and $P'(\cdot)$ refers to the corresponding probability when the input database is D'. The following definition is the main building block of this framework, which abstracts the key properties of our previous algorithms such as $\mathcal{S}$, $\mathcal{M}$ and $\mathcal{R}_S$:

Definition 24 ((ε, δ)-PTR function)**.** A function $T(D, s) : \mathcal{D} \times (\mathcal{C} \cup \{\perp\}) \mapsto \mathcal{C}' \cup \{\perp\}$ is called (ε, δ)-PTR if

1. $P(T = \perp | s = \perp) = 1$, for all D.

[16] Because the distribution of ϕ is continuous, the probability of having three or more lines intersecting at one point is 0.

2. For all $s \in C$, D and D' adjacent,

$$\begin{aligned} P(T = \perp | s) &\leq e^{\varepsilon} P'(T = \perp | s), \\ P(T \neq \perp | s) &\leq e^{\varepsilon} P'(T \neq \perp | s). \end{aligned} \tag{28}$$

3. There exists $G(T, D) \subseteq \mathcal{C}$, such that if $s \in G(T, D)$, then for all D' adjacent to D and all $C' \subseteq \mathcal{C}'$

$$P(T \in C' | s) \leq e^{2\varepsilon} P'(T \in C' | s), \tag{29}$$

4. if $s \notin G(T, D)$, then for all $D \in \mathcal{D}$, $P(T \neq \perp | s) \leq \delta$.

In our discussion, for notational simplicity we will drop the dependence on (ε, δ), by just saying PTR.

It is clear that when s does not depend on D, then a mechanism that computes the PTR function is $(2\varepsilon, \delta)$-differentially private. However, in the three examples we have seen, only the algorithm $\mathcal{S}$ falls into this category; in this case s is the bin width[17] In the other algorithms the input s, depends on D; indeed the inputs are themselves produced by a PTR function with D as part of its input. In addition there are still issues such as considering many discretizations, using the random partition as the input data base, etc. So we need a to consider a wider class of functions in the PTR framework.

Definition 25 (Cascade)**.** Let $t_1, \ldots, t_J$ be a sequence of elements in $\mathcal{C} \cup \{\perp\}$. The cascade function, Cas, applied to the sequence is the leftmost (that is, smallest indexed) t_j that is not $\perp$, $1 \leq j \leq J$, if such an element exists, and $\perp$ if no such element exists. That is,

$$\mathrm{Cas}_{j=1}^{J} t_j = t_{j_0},$$

where $j_0 = \min\{j \leq J : t_j \neq \perp\}$ and if $t_j = \perp$ for all j, $\mathrm{Cas}_{j \geq 1} t_j = \perp$. We call the number J the *length* of the cascade.

We abuse notation and, when there is no possibility of confusion, refer to a computation whose output is the result of applying a cascade operator over the outputs of a sequence of PTR computations as a cascade, or a cascade computation. Algorithm $\mathcal{S}$ is an example of a length 2 cascade, since up to two different discretizations are used in a computation, and the algorithm outputs the first non-$\perp$ value obtained. The input s is the bin width in the discretizations.

Sometimes the values in the sequence to which the cascade operator is applied are themselves the result of a (function of a) cascade. This is the case in the median Algorithm $\mathcal{M}$, whose output is a length 2 cascade where the elements in the sequence are computed using a value produced by Algorithm $\mathcal{S}$, itself a length 2 cascade computation. This suggests a cascade hierarchy, defined as follows.

Definition 26 (Level-K (ε, δ)-Cascade)**.** A function $V_K(D, s) : \mathcal{D} \times (\mathcal{C}_0 \cup \{\perp\}) \mapsto \mathcal{C}_K \cup \{\perp\}$ is a level-K cascade if

$$V_K(D, s) = \mathrm{Cas}_{j=1}^{J} T_K^{(j)}(D, V_{K-1}(D, s)),$$

where $V_{K-1} : \mathcal{D} \times (\mathcal{C}_0 \cup \{\perp\}) \mapsto \mathcal{C}_{K-1} \cup \{\perp\}$ is a level-$(K-1)$ (ε, δ)-cascade computation, all $T_K^{(j)}$, $1 \leq j \leq J : \mathcal{D} \times (\mathcal{C}_{K-1} \cup \{\perp\}) \mapsto \mathcal{C}_K \cup \{\perp\}$, are (ε, δ)-PTR functions, conditionally independent given the inputs, and $V_0(D, s) = s$.

[17] Strictly speaking, the bin width, being a function of $n = |D|$, actually depends on D. However, as noted above, in this work we assume n is known.

Note that in $V_K(D,s)$, there are $2\sum_{k=1}^K J_k$ (ε,δ)-PTR functions which are $(2\varepsilon,\delta)$-differentially private. Intuitively, $V_K(D,s)$ should be (ε',δ')-differentially private, with $\varepsilon' = 2\sum_{k=1}^K J_k\varepsilon$ and $\delta' = \sum_{k=1}^K J_k\delta$. This is true, by the next theorem:

Theorem 27 (Composition theorem for general (ε,δ)-differentially private algorithms)**.** *Let $T_1 : D \mapsto T_1(D)$ be (ϵ,δ)-d.p., and for all $J \geq 2$, $T_J : (D, s_1, \ldots, s_{J-1}) \mapsto T_J(D, s_1, \ldots, s_{J-1}) \in \mathcal{C}_J$ be (ϵ,δ)-d.p., for all given $(s_1, \ldots, s_{J-1}) \in \bigotimes_{j=1}^{J-1} \mathcal{C}_j$. Then for all neighboring D, D' and all $S \subseteq \bigotimes_{j=1}^{J} \mathcal{C}_j$*

$$P((T_1, \ldots, T_J) \in S) \leq e^{J\epsilon} P'((T_1, \ldots, T_J) \in S) + J\delta.$$

First note that in Theorem 27, for any j, the space $\mathcal{C}_j$ may contain $\perp$. It is not hard to see that the cascade composition $V_K(D,s)$ is a special case of the general composition. One just needs to arrange all the $T_k^{(j)}$ in a sequence $(T_1, \ldots, T_J)$, where $J = \sum_{k=1}^K J_k$, and $T_{J_1+\cdots+J_{k-1}+j} = T_k^{(j)}$ for all k and $1 \leq j \leq J_k$. Then $V_k(D,s)$ is just a function of $(T_1, \ldots, T_J)$, which indicates for any $C \subseteq \mathcal{C}_K \cup \{\perp\}$, $P(V_K(D,s)) \in C = P((T_1, \ldots, T_J) \in S_C)$ for some $S_C \subseteq \bigotimes_{k=1}^K (\mathcal{C}_k \cup \{\perp\})^{J_k}$.

As a result, we can directly apply Theorem 27 to get the the result that any level K (ε,δ)-cascade is (ε',δ')-differentially private, with $\varepsilon' = 2\sum_{k=1}^K J_k\varepsilon$ and $\delta' = \sum_{k=1}^K J_k\delta$.

A simple proof of Theorem can be given by induction, basing on the following lemma:

Lemma 28. *Let $T_1(D) : D \mapsto T_1(D) \in \mathcal{C}_1$ be an (ϵ,δ)-d.p. function, and for any $s_1 \in \mathcal{C}_1$, $T_2(D, s_1) : (D, s_1) \mapsto T_2(D, s_1) \in \mathcal{C}_2$ be an (ϵ,δ)-d.p. function given the second input s_1. Then we show that for any neighboring D, D', for any $S \subseteq \mathcal{C}_2 \times \mathcal{C}_1$, we have, using the notation in our paper*

$$P((T_2, T_1) \in S) \leq e^{2\epsilon} P'((T_2, T_1) \in S) + 2\delta. \tag{30}$$

Proof. For any $C_1 \subseteq \mathcal{C}_1$, define

$$\mu(C_1) = \left(P(T_1 \in C_1) - e^{\epsilon} P'(T_1 \in C_1)\right)_+,$$

then μ is a measure on $\mathcal{C}_1$ and $\mu(\mathcal{C}_1) \leq \delta$ since T_1 is (ϵ,δ)-d.p. . As a result, we have for all $s_1 \in \mathcal{C}_1$,

$$P(T_1 \in ds_1) \leq e^{\epsilon} P'(T_1 \in ds_1) + \mu(ds_1). \tag{31}$$

Also note that by the definition of (ϵ,δ)-d.p. , for any $s_1 \in \mathcal{C}_1$,

$$\begin{aligned} P((T_2, s_1) \in S) &\leq \left(e^{\epsilon} P'((T_2, s_1) \in S) + \delta\right) \wedge 1 \\ &\leq \left(e^{\epsilon} P'((T_2, s_1) \in S)\right) \wedge 1 + \delta. \end{aligned} \tag{32}$$

Then (31) and (32) give (30):

$$\begin{aligned} P((T_2, T_1) \in S) &\leq \int_{S_1} P((T_2, s_1) \in S) P(T_1 \in ds_1) \\ &\leq \int_{S_1} \left(\left(e^{\epsilon} P'((T_2, s_1) \in S)\right) \wedge 1 + \delta\right) P(T_1 \in ds_1) \\ &\leq \int_{S_1} \left(\left(e^{\epsilon} P'((T_2, s_1) \in S)\right) \wedge 1\right) P(T_1 \in ds_1) + \delta \\ &\leq \int_{S_1} \left(\left(e^{\epsilon} P'((T_2, s_1) \in S)\right) \wedge 1\right) \left(e^{\epsilon} P'(T_1 \in ds_1) + \mu(ds_1)\right) + \delta \\ &\leq e^{2\epsilon} \int_{S_1} P'((T_2, s_1) \in S) P'(T_1 \in ds_1) + \mu(S_1) + \delta \end{aligned}$$

$$\leq e^{2\epsilon} P'((T_2, T_1) \in S) + 2\delta. \tag{33}$$

In the equations above, S_1 denotes the projection of S onto $\mathcal{C}_1$. The event $\{(T_2, s_1) \in S\}$ refers to $\{(T_2(D, s_1), s_1) \in S\}$ (or $\{(T_2(D', s_1), s_1) \in S\}$). □

However, the special structure for (ε, δ)-PTR function and the cascade composition enables us to get better ε' and δ' for the differential privacy of (ε, δ)-cascade compositions. Note that in a level-K (ε, δ)-cascade composition, many $T_k^{(j)}$ may take value $\perp$, and with such an output, $T_k^{(j)}$ contributes $(\varepsilon, 0)$ rather than $(2\varepsilon, \delta)$ to the total bound of probability. A more careful investigation gives the next theorem:

Theorem 29. *A level-K (ε, δ)-cascade is (ε', δ')-differentially private, with $\varepsilon' = \left(K + \sum_{k=1}^{K} J_k\right)\varepsilon$, and $\delta' = \left(\sum_{k=1}^{K} J_k\right)\delta$, where J_k is the length of the cascade at level k.*

Roughly speaking, at each level there are at most J_k PTR functions, and their corresponding ε's add up, with all but one being simply ε and at most one being 2ε (according to $\perp$ or not). So the final privacy coefficient ε' is simply adding up the ε's at each level. The theorem says that the δ terms accumulate gracefully.

Before proving Theorem 29, we introduce some new notation. Since the randomness of an (ε, δ)-PTR computation (and hence a cascade) comes from only the coin tosses, independent of D and s, we write the probability of the random event $\{V(D, s) \in C\}$ as $P(C|s)$ whenever there is no confusion. Furthermore, suppose $V_K(D, s_0)$ is a level-K (ε, δ)-cascade with inputs D and s_0, and for any $C \subseteq \mathcal{C} \cup \{\perp\}$, and any sequence $(j_1, \ldots, j_K)$, $1 \leq j_k \leq J_k + 1$, $\forall\ k$, we write the probability of the event $\{T_k^{(j)} = \perp,\ \forall\ j < j_k,\ T_k^{(j_k)} \neq \perp,\ \forall\ k,\ V_K \in C\}$ as $P(j_1, \ldots, j_K, C|s_0)$. The case $j_k = J_k + 1$ means that $T_k^{(j)} = \perp$ for all j, and for convenience we let $T_k^{(J_k+1)} \equiv \perp$. Consequently we use the convention that $G(T_k^{(J_k+1)}, D)$ to be the whole space $\mathcal{C}_k \cup \{\perp\}$. As above, for convenience we use P' to denote the probability when the input is database D'.

Lemma 30. *If $V_K(D, s_0)$ is a level-K (ε, δ)-cascade, then for all $C \subseteq \mathcal{C} \cup \{\perp\}$, and any $1 \leq j_1 \leq J_1 + 1$,*

$$P(j_1, C|s_0) \leq e^{(K + \sum_{k=1}^{K} J_k)\varepsilon} P'(j_1, C|s_0) + \left(\sum_{k=2}^{K} J_k\right) \delta P(j_1|s_0) + \delta 1_{j_1 \leq J_1}.$$

Proof. We argue by induction. First, when $K = 1$,

$$\begin{aligned} & P(j_1, C|s_0) \\ =& P\left(T_1^{(j)} = \perp, \forall j < j_1, T_1^{(j_1)} \in C \,\middle|\, s_0\right) \\ =& P\left(T_1^{(j)} = \perp, \forall j < j_1 \,\middle|\, s_0\right) \int_C d(s, j_1; J_1) P(T_1^{(j_1)} \in ds|s_0), \end{aligned} \tag{34}$$

where the function $d(s, j; J)$ indicates the region of integration according to the value of j:

$$d(s, j; J) = \begin{cases} 1, & \text{if } s \neq \perp, \text{ and } j \leq J, \\ 1, & \text{if } s = \perp, \text{ and } j = J + 1, \\ 0. & \text{otherwise.} \end{cases}$$

By the assumption of PTR mechanisms, if $j_1 \leq J_1$ and $s_0 \notin G(T_1^{(j_1)}, D)$, then the above product is bounded by $P(T_1^{(j_1)} \neq \perp) \leq \delta$. Otherwise, letting $1_{j_1 \leq J_1}$ denote the indicator function

that takes value 1 if $j_1 \le J_1$ (in which case some non-$\perp$ value is returned by the cascade) and 0 if $j_1 = J_1 + 1$ (in which case the cascade has value $\perp$), we always have

$$d(s, j_1; J_1)P(T_1^{(j_1)} \in ds|s_0) \le e^{2\varepsilon 1_{j_1 \le J_1}} d(s, j_1; J_1)P'(T_1^{(j_1)} \in ds|s_0), \quad \forall s.$$

As a result,

$$\begin{aligned}
&P(j_1, C|s_0)\\
\le& e^{(j_1-1)\varepsilon} P'\left(T_1^{(j)} = \perp, \forall j < j_1 \middle| s_0\right) \int_C e^{2\varepsilon 1_{j_1 \le J_1}} d(s, j_1; J_1) P'\left(T_1^{(j_1)} \in ds \middle| s_0\right)\\
\le& e^{(J_1+1)\varepsilon} P'(j_1, C|s_0).
\end{aligned} \tag{35}$$

Therefore,

$$P(j_1, C|s_0) \le e^{(J_1+1)\varepsilon} P'(j_1, C|s_0) + \delta 1_{j_1 \le J_1},$$

i.e., the statement holds for $K = 1$.

Now assume $K \ge 2$ and the statement holds for $1, 2, \ldots, K-1$, then from the induction assumption we have:

$$\begin{aligned}
&P(C|V_1 = s)\\
=& \sum_{j_2=1}^{J_2+1} P(j_2, C|V_1 = s)\\
\le& \sum_{j_2=1}^{J_2+1} \left\{ e^{\left(K-1+\sum_{k=2}^K J_k\right)\varepsilon} P'(j_2, C|V_1 = s) + \left(\sum_{k=3}^K J_k\right) \delta P(j_2|V_1 = s) + \delta 1_{j_2 \le J_2} \right\}\\
\le& e^{\left(K-1+\sum_{k=2}^K J_k\right)\varepsilon} P'(C|V_1 = s) + \left(\sum_{k=2}^K J_k\right) \delta.
\end{aligned} \tag{36}$$

Then we check the validity of the statement for K:

$$\begin{aligned}
&P\left(j_1, C|s_0\right)\\
=& P\left(T_1^{(j)} = \perp, \forall j < j_1 \middle| s_0\right) \int P(C|V_1 = s) d(s, j_1; J_1) P\left(T_1^{(j_1)} \in ds \,|s_0\right).
\end{aligned} \tag{37}$$

If $j_1 \le J_1$ and $s_0 \notin G(T_1^{(j_1)}, D)$, then the above product is bounded by δ. Otherwise we have, by (36) and the fact that $d(s, j_1; J_1) \le 1$,

$$\begin{aligned}
&\int P(C|V_1 = s) d(s, j_1; J_1) P\left(T_1^{(j_1)} \in ds \middle| s_0\right)\\
\le& \int \left(e^{\left(K-1+\sum_{k=2}^K J_k\right)\varepsilon} P'(C|V_1 = s) + \left(\sum_{k=2}^K J_k\right) \delta \right) d(s, j_1; J_1) P\left(T_1^{(j_1)} \in ds \middle| s_0\right)\\
\le& \int e^{\left(K-1+\sum_{k=2}^K J_k\right)\varepsilon} P'(C|V_1 = s) e^{2\varepsilon 1_{j_1 \le J_1}} d(s, j_1; J_1) P'\left(T_1^{(j)} \in ds \middle| s_0\right)\\
&+ \left(\sum_{k=2}^K J_k\right) \delta \int d(s, j_1; J_1) P\left(T_1^{(j_1)} \in ds \middle| s_0\right)
\end{aligned}$$

$$
\begin{aligned}
=&e^{(K-1+\sum_{k=2}^{K}+2\times 1_{j_1\le J_1})\varepsilon}\int P'(C|V_1=s)d(s,j_1;J_1)P'(T_1^{(j_1)}\in ds|s_0)\\
&+\left(\sum_{k=2}^{K}J_k\right)\delta\int d(s,j_1;J_1)P\left(T_1^{(j_1)}\in ds\Big|\, s_0\right). \qquad (38)
\end{aligned}
$$

Plugging inequality (38) into the right hand side of (37), we have, when $s_0\in G(T_1^{(j_1)},D)$,

$$
\begin{aligned}
&P\left(j_1,C|s_0\right)\\
\le&P\left(T_1^{(j)}=\perp,\forall j<j_1\Big|\, s_0\right)\\
&\quad\times e^{(K-1+\sum_{k=2}^{K}J_k+2\times 1_{j_1\le J_1})\varepsilon}\int P'(j_2,C|V_1=s)d(s,j_1;J_1)P'\left(T_1^{(j_1)}\in ds\Big|\, s_0\right)\\
&\quad+P\left(T_1^{(j)}=\perp,\forall j<j_1\Big|\, s_0\right)\left(\sum_{k=2}^{K}J_k\right)\delta\int d(s,j_1;J_1)P\left(T_1^{(j_1)}\in ds\Big|\, s_0\right)\\
\le&e^{(j_1-1)\varepsilon}P'\left(T_1^{(j)}=\perp,\forall j<j_1|s_0\right)\\
&\quad\times e^{(K-1+\sum_{k=2}^{K}J_k+2\times 1_{j_1\le J_1})\varepsilon}\int P'(C|V_1=s)d(s,j_1;J_1)P'\left(T_1^{(j_1)}\in ds\Big|\, s_0\right)\\
&\quad+\left(\sum_{k=2}^{K}J_k\right)\delta P(j_1|s_0)\\
\le&e^{(K+\sum_{k=1}^{K}J_k)\varepsilon}P'(j_1,C|s_0)+\left(\sum_{k=2}^{K}J_k\right)\delta P(j_1|s_0). \qquad (39)
\end{aligned}
$$

Finally, for all s_0 we have

$$
P(j_1,C|s_0)\le e^{(K+\sum_{k=1}^{K}J_k)\varepsilon}P'(j_1,C|s_0)+\left(\sum_{k=2}^{K}J_k\right)\delta P(j_1|s_0)+\delta 1_{j_1\le J_1},
$$

that is, the statement holds for K. □

Proof of Theorem 29. The statement of theorem 29 is obtained by summing the result of Lemma 30 over all $1\le j_1\le J_1+1$. □

By the composition Theorem 27, Theorem 29 can be generalized easily to obtain differential privacy of multiple independent cascades.

Definition 31 (Parallel Composition)**.** A random mechanism $W(D,\mathbf{s}):\mathcal{D}\times\bigotimes_{l=1}^{L}(\mathcal{C}_{0,l}\cup\{\perp\})\mapsto\bigotimes_{l=1}^{L}(\mathcal{C}_{K,l}\cup\{\perp\})$ is a level-K L-(ε,δ)-parallel composition, if

$$
W(D,\mathbf{s})=\left(V_{1,K}(D,s_1),V_{2,K}(D,s_2),\ldots,V_{L,K}(D,s_L)\right),
$$

where $\mathbf{s}=(s_1,\ldots,s_L)$, and $V_{l,K},\ (1\le l\le L)$ are level-K (ε,δ)-cascades conditionally independent given D and $\mathbf{s}$. We call L the number of components in the parallel composition.

Corollary 32. *A level-K L-(ε,δ)-parallel composition is (ε',δ')-differentially private, with $\varepsilon'=(LK+\sum_{l,k}J_{l,K})\varepsilon$, and $\delta'=\sum_{l,k}J_{l,k}\delta$.*

9 General input

Through the discussion in the previous section, one might note that in the definition of level-K (ε, δ)-parallel composition, the first input of $T_{l,k}^{(j)}$ (that is, the database, let us call it $D_{l,k,j}$) does not have to be the same for all values of (l, k, j). Such differences can arise in several natural ways. For example, at the beginning of the shortcut algorithm the inputs are partitioned into n/p disjoint sets of size p, and some computation is performed on each subset independently. This would look like an ordinary (non-PTR) subroutine call on n/p distinct databases (all are drawn from the original database).

Now let us take a close look at random partitions. Suppose $|D| = |D'| = n = mp + q,\ 0 \le q \le p-1$, and $D = \{x_1, \ldots, x_n\}$. The random partition $\pi_p(D, \sigma) = \{\pi_p^i(D, \sigma)\}_{i=1}^m$, where $\pi_p^i(D, \sigma) = \left\{x_{\sigma((i-1)p+1)}, \ldots, x_{\sigma(ip)}\right\}$, and $\sigma = (\sigma(1), \ldots, \sigma(n))$ is a random permutation of $(1, \ldots, n)$ generated inside the procedure π_p.

If D and D' are adjacent, i.e., they differ at only one individual, with out loss of generality, suppose $D' = \{x_1, \ldots, x_{n-1}, x_n'\}$. Clearly, for a given σ, $\pi_p(D, \sigma)$ and $\pi_p(D', \sigma)$ has the same number of elements and differ at no more than one element. Note that $\pi_1(D, \sigma) = D$ for all σ.

Of course, the databases in subroutine calls can be drawn more generally from the original D, and not only by partitioning. In the case of partitioning, if most of the partitions of D' look just like those of D; there will only be a difference in one of the partitions. In the more general case there may be more differences. For example, the databases in the subroutine calls may be independent random subsamples of the original database. In this case, if the subsampling is done without replacement, then for any fixed sequence of random coins, corresponding subroutine calls will have databases differing in at most one element when the original input is D' and not D.

This motivates the observation that the results of the previous section hold when the first input of $T_{l,k}^{(j)}$ becomes $D_{l,k,j}$, provided that $D_{l,k,j}$ and $D'_{l,k,j}$ are adjacent for all (l, k, j).

Definition 33 (Generalized cascade)**.** For a sequence of databases, $\mathbf{D} = (D_1, \ldots, D_K)$, and $s \in (\mathcal{C}_0 \cup \{\perp\})$, a function $GV_K(\cdot, \cdot) : (\mathbf{D}, s) \mapsto GV_K(\mathbf{D}, s) \in \mathcal{C}_K \cup \{\perp\}$ is a level-K generalized (ε, δ)-cascade, if

$$GV_K(\mathbf{D}, s) = \mathrm{Cas}_{j=1}^{J_K} T_K^{(j)}\left(D_K, GV_{K-1}(\mathbf{D}_{1:K-1}, s)\right),$$

where GV_{K-1} is a level-$(K-1)$ generalized (ε, δ)-cascade and $T_K^{(j)}$ are (ε, δ)-PTR functions conditionally independent given the inputs. Here, $\mathbf{D}_{1:K-1}$ denotes the submatrix of $\mathbf{D}$ consisting of columns 1 through $(K-1)$.

Similarly we can define generalized (ε, δ)-parallel composition. Let $\mathbf{D}_{l,1:K-1}$ denotes the submatrix of $\mathbf{D}$ consisting of the lth row and columns 1 through $(K-1)$.

Definition 34 (Generalized parallel composition)**.** Let $\mathbf{D} = (D_{l,k})_{1 \le l \le L, 1 \le k \le K}, D_{l,k} \in \mathcal{D}$ and $\mathbf{s} \in \bigotimes_{l=1}^L (\mathcal{C}_{0,l} \cup \{\perp\})$, a function $GW(\cdot, \cdot) : (\mathbf{D}, \mathbf{s}) \mapsto GW(\mathbf{D}, \mathbf{s}) \in \bigotimes_{l=1}^L (\mathcal{C}_{K,l} \cup \{\perp\})$ is a level-K generalized L-(ε, δ)-parallel composition if

$$GW(\mathbf{D}, \mathbf{s}) = \left(V_{l,K}(D_{l,K}, V_{l,K-1}(\mathbf{D}_{l,1:K-1}, s_l))\right)_{l=1}^L,$$

where $V_{l,K}$ are level-K generalized (ε, δ)-cascade conditionally independent given the inputs.

Corollary 35. *Let $\mathbf{D} = (D_{l,k})_{1 \le l \le L, 1 \le k \le K}$ and $\mathbf{D}' = (D'_{l,k})_{1 \le l \le L, 1 \le k \le K}$. Suppose $(D_{l,k},\ D'_{l,k})$ is adjacent for all (l, k). If $GW(\cdot, \cdot)$ is a level-K generalized L-(ε, δ)-parallel composition, then for*

any $\mathbf{s} \in \bigotimes_{l=1}^{L}(\mathcal{C}_{0,1} \cup \{\perp\})$ *and* $C \subseteq \bigotimes_{l=1}^{L}(\mathcal{C}_{K,l} \cup \{\perp\})$,

$$P(GW \in C) \leq e^{\left(LK+\sum_{l,k} J_{l,k}\right)\varepsilon} P'(GW \in C) + \sum_{l,k} J_{l,k}\delta.$$

Example 36 (Short-cut regression)**.** In the short-cut regression algorithm, suppose the random partition is given by $\pi_p(D, \sigma)$, where σ is generated by the procedure π_p, independently of everything else. Let β_i be the β determined by the data points in $\pi_p^i(D, \sigma)$, if any. Then in the short-cut regression algorithm, we have $L = p$, $K = 2$, $J_{l,k} = 2$, and $D_{l,k} = \{\beta_i\}_{i=1}^{m}$ for all l, k.

Conditioning on σ, $D_{l,k}$ and $D'_{l,k}$ are adjacent, then by corollary 35, we have

$$P(\mathcal{R}_S(D) \in C|\sigma) \leq e^{6p\varepsilon} P(\mathcal{R}_S(D') \in C|\sigma) + 4p\delta.$$

Summing over all possible σ, we conclude that the short-cut regression algorithm $\mathcal{R}_S$ is $(6p\varepsilon, 4p\delta)$-differentially private. It should be noted that here $\delta = \frac{1}{2}\left(\frac{n}{p}\right)^{-\varepsilon \ln\left(\frac{n}{p}\right)} \in \nu(n)$.

What about the regression algorithm $\mathcal{R}_H$? Note that the algorithm $\mathcal{R}_H$ consists of two steps, where the first step estimates the scale of the uncertainty of the regression coefficient given by the data, which is a level-1 2-(ε, δ)-parallel composition. The second step is a level-1 (ε, δ)-cascade given the scale estimation in the first step. Also note that the regression coefficients are not estimated coordinate-wise, but estimated as a single vector in $\mathbf{R}^p$. This is none of the situations we considered above. So we need to consider one more level of complication, that is, the sequential composition of generalized parallel compositions.

Suppose $t = 1, \ldots, T$, and $\mathcal{C}_{t,k,l}$ are general measurable spaces. For $\mathbf{s}_t \in \bigotimes_{l=1}^{L_t}(\mathcal{C}_{t,0,l} \cup \{\perp\})$ and set of databases $\mathbf{D}_{t,*,*}$ (defined below, with corresponding dimensionality), $GW_t(\cdot,\cdot) : (\mathbf{D}_{t,*,*}, \mathbf{s}) \mapsto GW_t(\mathbf{D}_{t,*,*}, \mathbf{s}) \in \bigotimes_{l=1}^{L_t}(\mathcal{C}_{t,K,l} \cup \{\perp\})$, a level-$K_t$ generalized L-(ε, δ)-parallel composition. Define $(\mathbf{D})_{t,l,k}$, with $\mathbf{D} = (D_{t,l,k})_{1\leq t\leq T, 1\leq l\leq L_t, 1\leq k\leq K_t}$, and let $T_{t,l,k}^{(j_{t,l,k})}(D_{t,l,k}, \cdot)$ be the $j_{t,l,k}$th PTR function at kth level in the lth component in GW_t, and the length of cascade in $V_{t,l,k}$ is $J_{t,l,k}$. Consider the nested subroutine composition of a sequence of generalized parallel compositions. That is, let $GW_1, \ldots, GW_T$ be a sequence of generalized L_t-(ε, δ)-parallel compositions, for $1 \leq t \leq T-1$, GW_{t+1} calls GW_t as the second input. Then, by general composition Theorem 27 we have the following theorem:

Theorem 37. *Assume* $\mathbf{D} = (D_{t,l,k})_{1\leq l\leq L_t, 1\leq K\leq K_t}$ *and* $\mathbf{D}' = \left(D'_{t,l,k}\right)_{1\leq l\leq L_t, 1\leq K\leq K_t}$ *are two sets of data bases. If* $D_{t,l,k}$ *and* $D'_{t,l,k}$ *are adjacent for all* (t, l, k)*, then for any* $\mathbf{s} \in \bigotimes_{l=1}^{L_1}(\mathcal{C}_{1,0,l} \cup \{\perp\})$ *and* $C \subseteq \bigotimes_{l=1}^{L_T}(\mathcal{C}_{T,K_T,l} \cup \{\perp\})$,

$$P(C|\mathbf{s}) \leq e^{\varepsilon'} P'(C|\mathbf{s}) + \delta',$$

with $\varepsilon' = \sum_t \left(L_t K_t + \sum_{l,k} J_{t,l,k}\right)\varepsilon$*, and* $\delta' = \sum_{t,l,k} J_{t,l,k}\delta$.

Example 38 (Robust regression)**.** The regression algorithm $\mathcal{R}_H$ consists of two steps, first a scale estimation on the set of intersections determined by a random partition of D, then the estimated regression coefficient based on the scale estimation. The first step itself is a level-1 generalized p-(ε, δ)-parallel composition, with $D_{1,l,1} = \pi_p(D, \sigma)$, where σ is the random permutation independent of everything else.

The second step of regression algorithm $\mathcal{R}_H$ is the test and release of the regression coefficient, which is not done coordinate-wise. So this step is not a parallel composition but a level-1 (ε, δ)-cascade, with $\mathcal{C} = (\mathbf{R} \cup \{\perp\})^p$ and output in $\mathbf{R}^p \cup \{\perp\}$, and the length of cascade is 2^p. Summing

up, we have $K_1 = 1, L_1 = p, j_{1,l,k} = 2$, and $K_2 = 1, L_2 = 1, j_{2,l,k} = 2^p$. As a result, by theorem 37 we know that the regression algorithm $\mathcal{R}_H$ is $((2^p + 3p + 1)\varepsilon, (2p + 2^p)\delta)$-differentially private. Similarly, here $\delta = \frac{1}{2}\left(\frac{n}{p}\right)^{-\varepsilon \ln\left(\frac{n}{p}\right)} \in \nu(n)$.

10 Conclusions and Open Problems

The purpose of this work was two-fold: to obtain accurate, privacy-preserving, statistical estimators, and to test the theory that robust estimators form a good starting point for differential privacy. The outcome was uniformly positive: we succeeded in every attempt to adapt a robust estimator to obtain differential privacy, and the distortion always vanishes as n grows.

We remark that this work began with a median finding algorithm inspired by the fact that the empirical influence function for the sample median converges to a distribution if the underlying distribution F has positive density at $F^{-1}(1/2)$. The algorithm privately tests density of the empirical distribution at the sample median; if the density is sufficiently high, the algorithm proceeds; otherwise it outputs $\perp$. We eventually found a simpler algorithm, $\mathcal{M}$, described above.

The density test initiates a study of privacy-preserving hypothesis testing, an excellent topic for future research.

Acknowledgement

The potential connection between robustness and privacy was suggested by Werner Steutzle, before the invention of differential privacy. We warmly thank him for this contribution.

References

[1] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: A holistic solution to contingency table release. In *Proceedings of the 26th Symposium on Principles of Database Systems*, pages 273–282, 2007.

[2] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: The SuLQ framework. In *Proceedings of the 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, June 2005.

[3] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *Proceedings of the 40th ACM SIGACT Symposium on Thoery of Computing*, 2008.

[4] J. M. Borwein and A. S. Lewis. *Convex analysis and nonlinear optimization, theory and examples*. Springer, 2006.

[5] C. Dwork. Differential privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP)(2)*, pages 1–12, 2006.

[6] C. Dwork. An ad omnia approach to defining and achieving private data analysis. In F. Bonchi, E. Ferrari, B. Malin, and Y. Saygin, editors, *Privacy, Security, and Trust in KDD, First ACM SIGKDD International (PinKDD), Revised Selected Papers*, volume 4890 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2007.

[7] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: privacy via distributed noise generation. In *Advances in Cryptology: Proceedings of EUROCRYPT*, pages 486–503, 2006.

[8] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Theory of Cryptography Conference*, pages 265–284, 2006.

[9] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *Proceedings of CRYPTO 2004*, volume 3152, pages 528–544, 2004.

[10] R. A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London, Series A*, 222:309–368, 1922.

[11] D. Freedman and P. Diaconis. On the histogram as a density estimator: l_2 theory. *Z. Wahrscheinlichkeitstheorie verw. Gebiete*, 57:453–476, 1981.

[12] F. Hampel, E. Ronchetti, P. Rousseeuw, and W. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. John Wiley, New York, 1986.

[13] P. Huber. *Robust statistics*. John Wiley & Sons, 1981.

[14] S. Kasiviswanathan, H. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? In *Proceedings of FOCS 2008*, 2008.

[15] J. Kiefer and J. Wolfowitz. On the deviations of the empiric distribution function of vector chance variables. *Transactions of the American Mathematical Society*, 87:173–186, January 1958.

[16] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual Symposium on Foundations of Computer Science*, 2007.

[17] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the 39th ACM Symposium on Theory of Computing*, pages 75–84, 2007.

[18] D. Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, 1984.

[19] A. Smith. Efficient, differentially private point estimators, 2008.