Differential Privacy and Robust Statistics

Cynthia Dwork
Microsoft Research
dwork@microsoft.com

Jing Lei*
Department of Statistics
University of California, Berkeley
jinglei@stat.berkeley.edu

ABSTRACT

We show by means of several examples that robust statistical estimators present an excellent starting point for differentially private estimators. Our algorithms use a new paradigm for differentially private mechanisms, which we call Propose-Test-Release (PTR), and for which we give a formal definition and general composition theorems.

Categories and Subject Descriptors

H.2.0 [**Information Systems**]: Database Management—*security, integrity and protection*; G.3 [**Mathematics of Computing**]: Probability and Statistics—*statistical computing*

General Terms

Algorithms, Security and Theory

1. INTRODUCTION AND BACKGROUND

Over the last few years a new approach to privacy-preserving data analysis, based on *differential privacy* [6, 4], has born fruit [7, 1, 6, 14, 13, 2]. 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. 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 been developed; in particular [6] show that for an analysis $f : D \to \mathbf{R}^k$ it is sufficient to add Laplacian noise to each of the k outputs that is calibrated to the *sensitivity* of f, roughly, the worst-case over all D of the amount by which the data of any single individual can change the output of f. Many analyses can be formulated as insensitive functions, permitting high-quality differentially private results. However, the design of insensitive algorithms can require considerable re-thinking of existing algorithms.

*Much of this work was done while this author was supported by Microsoft Research.

STOC'09, May 31–June 2, 2009, Bethesda, Maryland, USA.

In this work we turn to a field in which considerable care has already been given to ensuring some kind of probabilistic insensitivity: *robust statistics*. We focus on *parameter estimation*, a branch of statistics that assumes data come from a parameterized family of probability distributions and makes inferences about the parameters of the distribution. Most well-known elementary statistical methods are parametric. For example, given data samples $x_1, \ldots, x_n$ drawn from a normal distribution $F \in \mathcal{F} = \{\mathcal{N}(\theta, 1)\}_{\theta \in \mathbf{R}}$, the goal is to estimate θ and a simple method averages the samples. Robust statistics is concerned with resilience against outliers and small errors in data measurement. A more robust method for estimating θ is to compute the sample median. More generally, robust statistics recognizes that real life does not match ideal conditions, and even the "best" distribution in $\mathcal{F}$ is only an approximation to the distribution underlying real data. We show by means of several examples that robust statistical estimators present an excellent starting point for differentially private estimators.

1.1 Robust Estimators and The Influence Function

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. 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, and is denoted $T(F)$. As a result, a statistical estimator can be viewed as a functional mapping the space of distribution functions to Euclidean space [9]. For example, suppose $X \in \mathbf{R}^1$ and Formally, robustness is captured via the *influence function* $\mathrm{IF}(x, T; F)$, an asymptotic notion describing how the estimator T applied to a distribution F changes if we introduce an infinitesimal contamination at x:

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

Typically, a robust estimator has bounded influence function [11, 9].

1.2 From Statistical Robustness to Privacy Always: Propose-Test-Release

Theorems about robustness typcially assume 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. The resulting claims of insensitivity are therefore probabilistic in nature. On the other hand, to ensure privacy always, 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 algorithm proposes a bound on sensitivity, either working with a default proposal guided by the proof of robustness or obtaining the proposal by engaging in preliminary differentially private computations; it then tests the adequacy of the proposed bound, again in a privacy-preserving fashion. If the response indicates high sensitivity, the algorithm outputs "$\perp$" and halts[1]. Since this decision is made based on the outcome of a differentially private test, no information is leaked by the decision itself. If the response indicates 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. We call this the *Propose-Test-Release* (PTR) paradigm.

Because our algorithms may halt, we need to show that in the statistical setting the algorithms not only produce answers, but in fact produce very accurate answers. For this, we again rely on the robustness of the estimators. The intuition for high accuracy is as follows. 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 – how much the value of the estimator can change if a single datum is modified – 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.

1.3 Additional Related Work

Nissim, Raskhodnikova, and Smith exploit low local sensitivity to improve accuracy of differentially private analyses in favorable cases [14]. They employ an insensitive method of upper bounding the local sensitivity and then add noise calibrated to the computed bound. Their algorithms always yield a response. However, their techniques can be surprisingly non-robust, yielding large noise even in some ideal cases. For example, in releasing the median a single exponentially far outlier may cause noise exponential in n, even in databases with 0 local sensitivity.

We have recently learned of the slightly older work of Heitzig, which uses a procedure inspired by the Quenouille-Tukey Jackknife technique to estimate local sensitivity, and then publishes either a correspondingly large *range* of values containing the true answer or a random perturbation scaled to this quantity [10]. No formal privacy guarantees are provided and composition is not addressed.

In parallel with our efforts, Smith [16] showed that for well-behaved parametric probability models one can construct an efficient (in the statistical sense) and unbiased estimator whose distribution converges to that of the maximum likelihood estimator.

The approach to robustness based on influence functions is due to Huber [11]; see also [9].

[1]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, and there is no point in insisting on an outcome.

2. DEFINITIONS

A database is a set of *rows*. We say databases D and D' are *adjacent*, or are *neighbors*, if their Hamming distance is 1.

DEFINITION 1. *[5] A randomized function $\mathcal{K}$ gives (ε, δ)-*differential privacy *if for all pairs of adjacent databases D and D', and all $S \subseteq Range(\mathcal{K})$,*

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

The probabilities are 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*. In the special case that $\delta = 0$ we say the algorithm has ε-differential privacy.

DEFINITION 2. *For $f : \mathcal{D} \to R^d$, the* L_1-sensitivity *of f is $\Delta f = \max_{D,D'} \|f(D) - f(D')\|_1$, where D and D' are neighbors.*

The Laplacian distribution with parameter b, denoted $\mathrm{Lap}(b)$, has probability density function $p(x) = \exp(-|x|/b)/2b$ and cumulative distribution function $D(x) = (1/2)(1+\mathrm{sgn}(x)(1-\exp(|x|/b)))$.

THEOREM 3 ([6]). *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.*

Additional Notation.

We use the following notation and terms.

$\mathcal{C}$ (also $\mathcal{C}'$): some general measurable space containing the range of the query function.[2] Usually we can think of $\mathcal{C}$ as $\mathbf{R}^d$ for some integer d, e.g., in our examples of scale, median, regression, *etc.* The notation $\mathcal{C}$ ($\mathcal{C}'$) may refer to different spaces in different expressions.
D and D': a pair of adjacent databases.
$\mathcal{D}$: the space of all databases.
F: a cumulative distribution function.
n: the size of database D, assumed to be publicly known.
"Change a data point" means modifying the value of a data point, keeping n fixed.
The notation $||x||$ denotes the L_2 norm of the vector x.

In our results there are two sources of randomness, the coin flips made by the algorithms, and randomness in the generation of the database. In the privacy arguments we always treat the databases 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

[2]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.

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 statements about privacy will be in terms of $P(\cdot)$ and $P'(\cdot)$.

The second source of randomness is the randomness in creating the database. We use $P_F(\cdot)$ to denote the probability of an event over samples from F. 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. $\tilde{P}(\cdot)$ refers to the probability considering both sources of randomness. Our statements about utility will be in terms of $\tilde{P}(\cdot)$.

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 $a_n/b_n = O_P(1)$.

3. THE SCALE

Our algorithm for data scale (dispersion) is the fundamental building block on which all our algorithms rest, and its analysis yielded the seeds of the Propose-Test-Release framework.

The interquartile range (IQR) [8] is a well-known robust estimate for the scale of the data. Consider the following rough intuition. Suppose the data are i.i.d. samples drawn from a distribution F. Then $\mathrm{IQR}(F)$, defined as $F^{-1}(3/4) - F^{-1}(1/4)$, 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 high at the two quartiles, then given enough samples from F the sample interquartile distance should be close to $\mathrm{IQR}(F)$.

At a high level, the algorithm first tests how many points need to be changed to obtain a data set with a "sufficiently different" interquartile distance. Only if the (noisy) reply is "sufficiently large" will the algorithm release an approximation to the interquartile range of the dataset. The definition of "sufficiently different" is multiplicative as an additive notion for difference of scale makes no sense – what would be the right scale for the additive amount? The algorithm therefore works with the logarithm of the scale. This leads to a multiplicative noise on the IQR. However, the accuracy can be improved, as we will see later, since any quantile (which therefore includes the IQR) can be released with small additive noise after this coarse IQR estimate. For the base of the logarithm we choose $1 + 1/\ln n$, as we now explain. Let $\mathrm{IQR}(D)$ denote the sample interquartile range when the the data set is D. If the data are drawn i.i.d. from a 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 $\ln(\mathrm{IQR}(D)) - \ln(\mathrm{IQR}(F)) = O_{P_F}(\frac{1}{\sqrt{n}})$.[3] 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. 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.[4]

To test whether the magnitude of noise is sufficient for differential privacy, we *discretize* $\mathbf{R}$ 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 $H_n(\hat{D}) \notin B_{k_1}^{(1)}$?

Let $A_0(D)$ be the true answer to Q_0. If $A_0(D) \geq 2$, then neighbors D' of D satisfy $|H_n(D') - H_n(D)| \leq 1$; that is, they are close to each other; however they may not be in the same interval in the discretization, for example, if $H_n(D)$ lies close to one of the endpoints of the interval $[k_1, k_1 + 1)$. Letting $R_0 = A_0(D) + \mathrm{Lap}(1/\varepsilon)$, a small R_0 might indicate high sensitivity of the interquartile range. To cope with the case that a small R_0 is encountered only because of the boundary problem just described, we consider 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.

Algorithm $\mathcal{S}(D, n, \varepsilon)$:

1. For the jth discretization ($j = 1, 2$):
a. Compute $R_0(D) = A_0(D) + z_0$, where $z_0 \in_R \mathrm{Lap}(1/\varepsilon)$.[5]
b. 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 $z_s^{(j)} \sim \mathrm{Lap}(1/\varepsilon)$.
2. If $s^{(1)} \neq \perp$, return $s^{(1)}$; Otherwise return $s^{(2)}$.

The algorithm can be optimized by only computing $s^{(2)}$ if $s^{(1)} = \perp$. The algorithm has a special form, which we call a *cascade* (several computations are performed and the output is the first non-$\perp$ result). We discuss this further in Section 5.

THEOREM 4. *(a) Algorithm $\mathcal{S}$ is $(3\varepsilon, n^{-\varepsilon \ln n})$-differentially private. (b) Assuming the data are sorted Algorithm $\mathcal{S}$ runs in $O(n)$ time. (c) If $D = (X_1, ..., X_n)$, where $X_i \overset{iid}{\sim} F$ and 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 $\alpha > 0$,*

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

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

[3]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 $y = \mathrm{IQR}(D)$.

[4]Actually $\Omega(n^{-1/2+\gamma})$ with some small $\gamma > 0$ would work, here we just explain the feasibility but not focus on the optimality of the magnitude of noise. This is also true for our median and regression algorithms.

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

PROOF. (Sketch.) **(a): Privacy.** There are two interesting parts to the proof of privacy. First, letting s be shorthand for the result obtained with a single discretization, and defining $\mathcal{D}_0 = \{D : A_0(D) \geq 2\}$, we show that for all $C \subseteq \mathbf{R}^+$ and $D \in \mathcal{D}_0$, $P(s \in C) \leq e^{2\varepsilon} P'(s \in C)$. It follows that if $D \in \mathcal{D}_0$, then $\forall s_0 \in \mathbf{R}^+$ $P(s \in ds_0) \leq e^{2\varepsilon} P'(s \in ds_0)$. The algorithm tests that D is "deep" inside $\mathcal{D}_0$, so the threat to privacy is an erroneously large $R_0 = A_0(D) + \text{Lap}(1/\varepsilon)$, which occurs with probability at most $\delta = \frac{1}{2} n^{-\varepsilon \ln n}$. Thus, we get $(2\varepsilon, \delta)$ privacy for each discretization. A general composition result for (ε, δ)-differential privacy (Theorem 16) immediately yields $(4\varepsilon, 2\delta)$ privacy; however, the special form of Algorithm $\mathcal{S}$ as a cascade can be exploited to yield the smaller bound (Theorem 17).

(c): Good Behavior in Statistical Settings. 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 $j = 1, 2$. Since F is differentiable with positive derivatives at the two quartiles 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) $\xi n^{2/3} > 4 \ln^2 n + 4$; and (4) $\left(\frac{r'_2 - l'_1}{l'_2 - r'_1}\right)^4 < 1 + \frac{1}{\ln n}$. Intuitively, (4) is reasonble because $(\text{IQR}(F) + 4n^{-1/3})/(\text{IQR}(F) - 4n^{-1/3})$ is very close to 1; more to the point it ensures that

$$\log_{1+1/\ln n}(\text{IQR}(F)+4n^{-1/3}) - \log_{1+1/\ln n}(\text{IQR}(F)-4n^{-1/3})$$

is less than $1/4$, whence the two logarithms will lie in the same bin in at least one of the discretizations.

We consider the following two random (over the draw of D from F) events, $E_1 = \{q_1(D) \in (l_1, r_1), q_2(D) \in (l_2, r_2)\}$ and $E_2 := \{\rho \geq \frac{1}{2}\xi n^{2/3}\}$, where $\rho := \min_j \min\{|D \cap (l'_j, l_j)|, |D \cap (r_j, r'_j)|\}$. We argue that both events occur with all but negligible in n probability (over n random draws of F); the argument for E_1 uses a well known result about the deviations of the empirical distribution (see [12]), and that for E_2 relies on Hoeffding's inequality.

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 $1 - c_1 e^{-c_2 n^{1/3}}$.

Since at least one of $B^{(1)}_{k_1}$ and $B^{(2)}_{k_2}$ well covers $H_n(F)$, for at least one discretization, we have $A_0(D) \geq 2\ln^2 n + 2$, with probability at least $1 - c_1 e^{-c_2 n^{-1/3}}$. Thus 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). Thus we have the desired result: $\tilde{P}(\mathcal{S}(D) = \perp) = \tilde{P}(s_1(D) = \perp \text{ and } s_2(D) = \perp) = O(n^{-\varepsilon \ln n})$. The claim $\mathcal{S}(D) - IQR(F) \xrightarrow{\tilde{P}} 0$ follows from consistency of sample quantiles (*i.e.*, that $|IQR(D) - IQR(F)|$ converges to 0 in probability), and the fact that $(1 + \frac{1}{\ln n})^{z_s^{(j)}} \xrightarrow{P} 1$, for $j = 1, 2$.

(d): Rate of Convergence.

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

□

Trimmed Mean and Median.

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

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

The differentially private version of the α-trimmed mean is obtained easily from Algorithm $\mathcal{S}$, modified to find the α interquantile range. Assuming this has been done, and the value s_α has been returned, the value returned is

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

where z is a random draw from Lap(1) and $\kappa \in (0, 1)$ is a parameter. It follows from the properties Algorithm $\mathcal{S}$ that with probability $1 - O(n^{-cn^{1/3}})$ we have $s_\alpha n^\kappa \geq IQR_\alpha(D)$. In the statistical setting, the additive noise in the estimated α-trimmed mean is of order $O_{P_F}(n^{-1+\kappa})$.

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 using Algorithm $\mathcal{S}$. If Algorithm $\mathcal{S}$ returns $\perp$ then we output $\perp$. Otherwise, we discretize the line with bins of width $h = sn^{-1/3}$ (or $h = sn^{-1/2+\gamma}$ for some small $\gamma > 0$), and ask how much the database must change in order to drag the median out of its current bin. If the answer is less than $\ln^2 n + 2$, 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 discretization shifted by $h/2$.

THEOREM 5. *(1) Algorithm $\mathcal{M}$ is $(6\varepsilon, \nu(n))$-differentially private. (2) The computation cost for $\mathcal{M}$ is $O(n)$ assuming the data are sorted. (3) Under the conditions in theorem 4 (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

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

This algorithm works for general quantiles including those needed to compute any interquantile range. Thus, we can get a better interquartile range by first using Algorithm $\mathcal{S}$ to get a coarse estimate of the data scale, and using the output to compute the locations of the lower and upper quartiles.

4. 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 i.i.d. samples from the joint distribution of (X, Y), and the inference task is to estimate β^*, the true value of the model parameter 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.

4.1 A Short-Cut Regression Method

The short-cut regression algorithm, $\mathcal{R}_S$, similar to an algorithm proposed by Siegel (see Section 6.4.1 of [9]), is also reminiscent of the Subsample-and-Aggregate framework of Nissim *et al.* [14]. 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 describe 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. E.g., for the block $\{(x_1, y_1), (x_2, y_2)\}$, the corresponding $\beta = \mathbf{X}^{-1}\mathbf{Y} = \beta^* + \mathbf{X}^{-1}\mathbf{\Phi} = \beta^* + \tilde{\phi}$, where $\mathbf{X} = (x_1, x_2)^T$, $\mathbf{Y} = (y_1, y_2)^T$, $\mathbf{\Phi} = (\phi_1, \phi_2)^T$. This gives $n/2$ independent approximations to each coordinate of β^*. For each coordinate β_i^*, $i = 1, 2$, run Algorithm $\mathcal{M}$ on a dataset consisting of the $n/2$ different approximations for β_i^*, to obtain a single output value $\hat{\beta}_i$ for this coordinate. The output of Algorithm $\mathcal{R}_S$ is the vector $(\hat{\beta}_1, \hat{\beta}_2)$.

4.2 A Robust Regression Estimator

In this subsection we follow the Propose-Test-Release framework: 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 robust regression estimator $\mathbf{H}$ whose output $\hat{\beta}$ is given by

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

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

For ease of exposition we consider the case $p = 2$; the generalization to other values 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 (1). 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 correspond to the two coordinates of β. As a result, $\mathbf{R}^2$ is discretized into rectangular 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 each discretization $C^{(j)}$, $j = 1, \ldots, 4$, 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 do we 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' we have $|\beta_d(D') - \beta_d(D)| \leq h_d$ for $d = 1, 2$.

Algorithm $\mathcal{R}_H(D, n, \varepsilon)$.

[1.] Compute $\beta(D)$.

[$2-4$.] Partition the data into $n/2$ sets; within each group compute β as in the short-cut algorithm; define $D^{(d)}, d = 1, 2$ to be the set containing the dth coordinates of the $n/2$ β's.

[5.] For $1 \leq d \leq 2$, 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, $1 \leq d \leq 2$, and let $h_d = s_d/n^{1/4}$. Also, if $s_d = 0$, let $h_d = n^{-1/2}$.

[6.] For $1 \leq j \leq 4$, compute $R_2^{(j)}(D, (h_1, h_2))$.

If $R_2^{(j)}(D, (h_1, h_2)) \leq \ln n^2 + 2$, 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 $\mathcal{R}_H(D) = \beta^{(j)}$, else return $\mathcal{R}_H(D) = \perp$.

THEOREM 6. *(a) The algorithm $\mathcal{R}_H$ is $(11\varepsilon, \nu(n))$-differentially private*[6]*. (b) $\mathcal{R}_H$ runs in time $T(p, n)+(n/p)T(p, p)+O(n^{p+1})$ time, where for all p, n $T(p, n)$ denotes the running time of the (non-private) optimization algorithm in p dimensions on data sets of size n. The interesting part of this claim is the $O(n^{p+1})$ term. (c) If for all $1 \leq d \leq p$, $\tilde{\phi}_d = \mathbf{X}^{-1}\mathbf{\Phi}_d$ has continuous and positive density; and $f(\beta) = E_F|Y - X^T\beta|/||X||$ 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.

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

REMARK 7. *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 6, 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).*

Proof of Theorem 6.

Part (a) follows from the general proof in Section 5.

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

$\mathbf{Q}_\beta$: 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 (1) 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 (1) 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 discussion below, the terms "region", "line segment", "half line" and "intersection" all refer to those defined by the ℓ_i, $i = 1, \ldots, n$.

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

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

DEFINITION 8 (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 9 ([3]). *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 10. *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 11. $A_\beta(D) = \lceil \inf_{\gamma \in \partial f_D(\beta)} ||\gamma|| \rceil$.

PROOF. Suppose β is surrounded by r_0 regions[7], 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 from D by k additions and deletions, and the corresponding objective function in (1) 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 set.

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

$$0 = \sum_{t=1}^{t_0} \mu_t(\gamma_{r_t} + \eta_t) \Rightarrow \sum_{t=1}^{t_0} \mu_t \gamma_{r_t} = -\sum_{t=1}^{t_0} \mu_t \eta_t$$

$$\Rightarrow \sum_{r=1}^{r_0} \mu'_r \gamma_r = -\sum_{t=1}^{t_0} \mu_t \eta_t, \qquad \mu'_r \geq 0, \quad \sum_r \mu'_r = 1$$

$$\Rightarrow \left|\left| \sum_{r=1}^{r_0} \mu'_r \gamma_r \right|\right| = \left|\left| \sum_{t=1}^{t_0} \mu_t \eta_t \right|\right| \leq \max_t ||\eta_t||$$

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

$$\Rightarrow \max_t ||\eta_t|| \geq \inf_{\gamma \in \partial f(\beta)} ||\gamma||.$$

The second implication follows by adding coefficients of the γ_{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 adding or deleting k data points in D, and that the magnitude of change on the subgradient by a single addition or deletion is at most 1 (because in $f_D(\beta)$ the ith term is normalized by $||x_i||$). 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 the solution of (1), one needs to add or delete 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 $\partial f_{\hat{D}}(\beta) = \partial f_D(\beta) - \gamma_0$ (a translation by γ of $\partial f_D(\beta) - \gamma_0$), making β the solution of (1). To see this, let $k = \lceil ||\gamma_0|| \rceil$. 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 $0 \in \partial f_{\hat{D}}(\beta)$.
Thus, $A_\beta(D) \leq \left\lceil \inf_{\gamma \in \partial f_D(\beta)} ||\gamma|| \right\rceil$. $\square$

To complete the proof of part (b) of Theorem 6, 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 identify 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 6 (good behavior on nice distributions) . The main idea is the same as previous: use law of large numbers to ensure, with high probability, that $A_\beta(D)$ 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_D 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 6, approximates $A_\beta(D)$ within a constant of 2. This is because there are at most two lines crossing β, which indicates $\text{diam}(\partial f_D(\beta)) \leq 2$. To see this, consider a β at the intersection of two of the lines ℓ_i, $i = 1 \ldots, n$. Then f_D, the object function of the optimization problem, is not differentiable at β, and $\partial f_D(\beta)$ is a parallelogram where each edge is of length 1. So, the diameter of $\partial f_D(\beta)$ is at most 2.

It will therefore suffice to show that $\min_{\beta \notin C^{(j)}(D)} ||g(\beta)|| \gg \ln^2 n$ for some j, with high probability. Recall that

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

then our assumptions of twice differentiablity and positive definiteness of $E_F X X^T / ||X||$ ensures that f has a unique minimum β^*.

Consider the cell L and event E_0 defined by:

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

$$E_0 = \left\{ s_d \in \left[\frac{1}{2} n^{-1/20} IQR(\tilde{\phi}_d), 2n^{1/20} IQR(\tilde{\phi}_d)\right], 1 \leq d \leq p \right\},$$

It follows easily from Theorem 4 that $\tilde{P}(E_0) \geq 1 - O(n^{-c_1 \ln n})$. Assuming E_0, we have $||h|| \to 0$, by the assumptions that f is twice continuously differentiable and $E_F X X^T / ||X||$ is positive definite, there exists constant c such that for large enough n,

$$||\nabla f(\beta)|| \geq c||h||, \quad \text{for all } \beta \in L^C. \tag{2}$$

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 $||A_\beta(D)|| = \Omega(nh)$. Formally, 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

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

and as a result, by Lemma 11,

$$A_\beta(D) \geq n||g(D; \beta)|| - 1, \quad \text{for all } \beta \in \left(\bigcup_{i=1}^n \ell_i\right)^C. \tag{3}$$

We will first 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 will extend the result to all $\beta \in L^C$, using the continuity of the distribution of ϕ and X.

Let B be a countable dense subset of $L^C \cap \left(\bigcup_{i=1}^n \ell_i\right)^C$. and for all β and for $d = 1, 2$, let $\nabla_d f(\beta)$, respectively $g_d(D; \beta)$ denote the dth coordinate of $\nabla f(\beta)$ and $g(D; \beta)$. The theory of empirical processes gives us the following lemma, whose proof follows largely the argument used in [15, Ch II.3]:

LEMMA 12. *For $d = 1, 2$*

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

Let E_1 be the event $\left\{\sup_{\beta\in B}|g_d(D;\beta)-\nabla_d f(\beta)|\le n^{-1/3}\right\}$. Then $P_F(E_1)\ge 1-O\left(n^2\exp(-c_1 n^{1/3})\right)$.

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

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

Since B is dense in L^C, the same bound holds for all $\beta\in\left(\left(\bigcup_{i=1}^n\ell_i\right)\bigcup L\right)^C$, i.e.,

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

Furthermore, for those $\beta\in\ell_i\cap L^C$ for some i, since there are at most two lines crossing β,[8] based on the fact that $\mathrm{diam}(\partial f_D(\beta))\le 2$ and that B is dense in L^C, we have

$$A_\beta(D)\ge\inf_{\beta\in\left(\left(\cup_{i=1}^n\ell_i\right)\cup L\right)^C}A_\beta(D)-2.$$

Then we finally have

$$\inf_{\beta\in L^C}A_\beta(D)\ge\Omega\left(n^{7/10}\right). \tag{4}$$

Clearly (4) implies that $\beta(D)\in L$ because $A_{\beta(D)}(D)=0$. Furthermore, by the construction of $C^{(j)}(D), j=1,\dots,4$, and the definition of L, there exists at least one j, such that $L\subset C^{(j)}(D)$. Then (4) implies $A_2^{(j)}(D)\ge\Omega\left(n^{7/10}\right)$. So for large enough n, we have $A_2^{(j)}(D)\ge 2\ln^2 n+2$, which implies $\tilde{P}(\mathcal{R}_H(D)=\perp|E_0\cap E_1, s\ne\perp)=O(n^{-\varepsilon\ln n})$. But as shown in Theorem 4(c), $\tilde{P}(s\ne\perp)\ge 1-O(n^{-c_1\ln n})$ under our assumptions, as a result, we have the desired inequality for theorem 6 (c):

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

5. (ε,δ)-PTR FUNCTIONS

We have seen several examples of (ε,δ)-differentially private robust estimators that share the same spirit: first an insensitive magnitude is proposed, then we test privately whether this magnitude is big enough such that an additive Laplacian random noise calibrated to it is enough to provide the ε-differential privacy. In this section we formally define Propose-Test-Release algorithms and prove composition properties for them.

Henceforth we consider (possibly 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. Alternatively, 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 an arbitrary neighboring database 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'.

DEFINITION 13 ((ε,δ)-PTR FUNCTION). *A function $T(D,s):\mathcal{D}\times(\mathcal{C}\cup\{\perp\})\mapsto\mathcal{C}'\cup\{\perp\}$ is (ε,δ)-PTR if*

1. *$P(T=\perp|s=\perp)=1$, for all D.*
2. *For all $s\in\mathcal{C}$, D and D' adjacent,*
$$\begin{aligned}P(T=\perp|s)&\le e^{\varepsilon}P'(T=\perp|s),\\ P(T\ne\perp|s)&\le e^{\varepsilon}P'(T\ne\perp|s).\end{aligned}\tag{5}$$
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 $C'\subseteq\mathcal{C}'$*
$$P(T\in C'|s)\le e^{2\varepsilon}P'(T\in C'|s),\tag{6}$$
4. *$\forall D\in\mathcal{D}$, if $s\notin G(T,D)$, then $P(T\ne\perp|s)\le\delta$.*

We will drop the "(ε,δ)" and just say "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[9]. 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. On the other hand, in some algorithms such as the short-cut regression algorithm, one needs to deal with multiple dimensions in parallel from the same database. We therefore need to understand the composition properties of PTR functions. We consider several types of composition: cascading, in which several functions are invoked and the first non-$\perp$ value is returned; subroutine calls; parallel PTR operations invoked on partitions of the database; and parallel PTR operations invoked on random subsamples drawn without replacement from the database.

The cascade operation on a sequence of values simply returns the first item in the sequence that is not $\perp$, if one exists.

DEFINITION 14 (CASCADE). *Let $t_1,\dots,t_J$ be a sequence of elements in $\mathcal{C}\cup\{\perp\}$. The cascade function, Cas, applied to the sequence is $Cas_{j=1}^J t_j=t_{j_0}$, where $j_0=\min\{j\le J: t_j\ne\perp\}$ and if $t_j=\perp$ for all j, $Cas_{j\ge1}t_j=\perp$. We call the number J the* length *of the cascade.*

We also use the terms "cascade" and "cascade computation" to refer to a computation whose output is the result of applying a cascade operator to the outputs of a sequence of PTR computations. Thus, 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.

Algorithm $\mathcal{M}$ is a length 2 cascade where the elements in the sequence are the outputs of Algorithm $\mathcal{S}$, itself a length 2 cascade computation. This type of sequential composition suggests the following hierarchy.

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

[9] Recall we assume n is known.

DEFINITION 15 (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*

$$V_K(D,s)=Cas_{j=1}^{J_K}T_K^{(j)}(D,V_{K-1}(D,s)),$$

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

The cascade does not correspond to a tree. A the bottom level (level 1) cascade produces a single value, which is an input to all computations at the cascade in the next level, which in turn produces a single value, and so on. Note that in $V_K(D,s)$, there are $\sum_{k=1}^K J_k$ (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 16 (COMPOSITION OF (ε,δ)-DP ALGS). *Let $T_1:D\mapsto T_1(D)$ be (ϵ,δ)-d.p., and for all $J\ge 2$, $T_J:(D,s_1,\ldots,s_{J-1})\mapsto T_J(D,s_1,\ldots,s_{J-1})\in\mathcal{C}_J$ be (ϵ,δ)-d.p., for all given $(s_1,\ldots,s_{J-1})\in\bigotimes_{j=1}^{J-1}\mathcal{C}_j$, where "$\bigotimes$" denotes direct product of spaces. Then for all neighboring D,D' and all $S\subseteq\bigotimes_{j=1}^{J}\mathcal{C}_j$*

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

Note that in Theorem 16, 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. However, the special structure of 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 17. *A level-K (ε,δ)-cascade is (ε',δ')-dp, 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.

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). For example, at the beginning of the shortcut algorithm the inputs are partitioned into n/p disjoint sets of size p, and a computation is performed on each subset independently. This would look like an ordinary (non-PTR) subroutine call for each of the n/p database partitions.

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 $[n]$ 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 $D_{l,k,j}$ can be drawn more generally from the original D, and not only by partitioning. For example, they 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.

In fact, 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 18 (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*

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

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 19 (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 20. *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 $C\subseteq\bigotimes_{l=1}^L(\mathcal{C}_{K,l}\cup\{\perp\})$,*

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

EXAMPLE 21 (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, so by Corollary 20,*

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

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

For Algorithm $\mathcal{R}_H$ we need to consider one more level of complication: 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

$\mathbf{D} = (D_{t,l,k})_{1 \le t \le T, 1 \le l \le L_t, 1 \le k \le K_t}$, 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 $1 \le t \le T-1$, GW_{t+1} calls GW_t as the second input.

THEOREM 22. *Assume* $\mathbf{D} = (D_{t,l,k})_{1 \le l \le L_t, 1 \le K \le K_t}$ *and* $\mathbf{D}' = \left(D'_{t,l,k}\right)_{1 \le l \le L_t, 1 \le K \le K_t}$ *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* $C \subseteq \bigotimes_{l=1}^{L_T} (\mathcal{C}_{T,K_T,l} \cup \{\perp\})$, $P(C|\mathbf{s}) \le e^{\varepsilon'} P'(C|\mathbf{s}) + \delta'$, *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 23 (ROBUST REGRESSION). *Algorithm* $\mathcal{R}_H$ *begins with a set of scale estimations, one for each coefficient in a* β *computed on a random partition of* D*. This 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. This is followed by the test and release of the regression coefficient, 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* $K_2 = 1, L_2 = 1, j_{2,l,k} = 2^p$*. Thus, by Theorem 22,* $\mathcal{R}_H$ *is* $((2^p + 3p + 1)\varepsilon, (2p + 2^p)(1/2)(n/p)^{-\varepsilon \ln(\frac{n}{p})})$*-dp.*

6. CONCLUSIONS

We have demonstrated, by means of several examples, that robust estimators are a useful starting point for constructing highly accurate differentially private estimators. We have also introduced the Propose-Test-Release paradigm for exploiting local sensitivity.

It would be nice to have a general theorem describing the conditions under which this approach is fruitful, either in general or in the context of Propose-Test-Release protocols, especially since, as we have seen, arguing ease of computation and good behavior under statistical assumptions can be quite involved. The general question of the difficulty of determining distance from high-sensitivity datasets is also interesting.

Acknowledgement.

Werner Stuetzle suggested a possible connection between robustness and private data analysis. We warmly thank him for this contribution.

7. REFERENCES

[1] 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.

[2] 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.

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

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

[5] 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.

[6] 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.

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

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

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

[10] J. Heitzig. The "jackknife" method: Confidentiality protection for complext statistical analyses. In *Joint UNECE/Eurostat work session on statistical data confidentiality*, 2005.

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

[12] 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.

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

[14] 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.

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

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