Differentially Private Simple Linear Regression

Daniel Alabi
Harvard John A. Paulson School of Engineering and Applied Sciences

Audra McMillan
Department of Computer Science, Boston University
Khoury College of Computer Sciences, Northeastern University

Jayshree Sarathy
Harvard John A. Paulson School of Engineering and Applied Sciences

Adam Smith
Department of Computer Science, Boston University

Salil Vadhan
Harvard John A. Paulson School of Engineering and Applied Sciences

ABSTRACT

Economics and social science research often require analyzing datasets of sensitive personal information at fine granularity, with models fit to small subsets of the data. Unfortunately, such fine-grained analysis can easily reveal sensitive individual information. We study algorithms for simple linear regression that satisfy *differential privacy*, a constraint which guarantees that an algorithm's output reveals little about any individual input data record, even to an attacker with arbitrary side information about the dataset. We consider the design of differentially private algorithms for simple linear regression for small datasets, with tens to hundreds of datapoints, which is a particularly challenging regime for differential privacy. Focusing on a particular application to small-area analysis in economics research, we study the performance of a spectrum of algorithms we adapt to the setting. We identify key factors that affect their performance, showing through a range of experiments that algorithms based on robust estimators (in particular, the Theil-Sen estimator) perform well on the smallest datasets, but that other more standard algorithms do better as the dataset size increases.

KEYWORDS

differential privacy, linear regression, robust statistics, small-area analysis

1 INTRODUCTION

The analysis of small datasets, with sizes in the dozens to low hundreds, is crucial in many social science applications. For example, neighborhood-level household income, high school graduation rate, and incarceration rates are all studied using small datasets that contain sensitive, and often protected, data (e.g., [11]). However, the release of statistical estimates based on these data quantities—if too many and too accurate—can allow reconstruction of the original dataset [15]. The possibility of such attacks led to differential privacy [17], a rigorous mathematical definition used to quantify privacy loss. Differentially private (DP) algorithms limit the information that is leaked about any particular individual by introducing random distortion. The amount of distortion, and its effect on utility, are most often studied for large datasets, using asymptotic tools. When datasets are small, one has to be very careful when calibrating differentially private statistical estimates to preserve utility.

In this work, we focus on simple (i.e. one-dimensional) linear regression and show that this prominent statistical task can have accurate differentially private algorithms even on small datasets. Our goal is to provide insight and guidance into how to choose a DP algorithm for simple linear regression in a variety of realistic parameter regimes.

Even without a privacy constraint, small sample sizes pose a problem for traditional statistics, since the variability from sample to sample, called the *sampling error*, can overwhelm the signal about the underlying trend. A reasonable concrete goal for a DP mechanism, then, is that it not introduce substantially *more* uncertainty into the estimate. Specifically, we compare the noise added in order to maintain privacy to the standard error of the (nonprivate) OLS estimate. Our experiments indicate that for a wide range of realistic datasets and moderate values of the privacy parameter, ε, it is possible to choose a DP linear regression algorithm that introduces distortion less than the standard error. In particular, in our motivating use-case of the Opportunity Atlas [12], described below, we design a differentially private algorithm that matches or outperforms the heuristic method currently deployed, which does not formally satisfy differential privacy.

1.1 Problem Setup

1.1.1 Simple Linear Regression. In this paper we consider the most common model of linear regression: one-dimensional linear regression with homoscedastic noise. This model is defined by a slope $\alpha \in \mathbb{R}$, an intercept $\beta \in \mathbb{R}$, a noise distribution F_e with mean 0 and variance σ_e^2, and the relationship

$$\mathbf{y} = \alpha \cdot \mathbf{x} + \beta + \mathbf{e}$$

where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. Our data consists of n pairs, (x_i, y_i), where x_i is the explanatory variable, y_i is the response variable, and each random variable e_i is draw i.i.d. from the distribution F_e. The goal of simple linear regression is to estimate α and β given the data $\{(x_1, y_1), \cdots, (x_n, y_n)\}$.

Let $\mathbf{x} = (x_1, \ldots, x_n)^T$, $\mathbf{y} = (y_1, \ldots, y_n)^T$. The Ordinary Least Squares (OLS) solution to the simple linear regression problem is

AS and AM were supported by NSF awards CCF-1763786 and IIS-1447700, a research award from the Sloan Foundation, and (for AM) the Hariri Institute for Computing. At Northeastern, AM was supported by a Fellowship from the Cybersecurity & Privacy Institute and NSF grant CCF-1750640. SV was supported by a Simons Investigator Award. DA was partially supported by a Fellowship from Facebook. SV, DA, JS, AM and AS were supported by Cooperative Agreement CB16ADR0160001 with the Census Bureau. The views expressed in this paper are those of the authors and not those of the U.S. Census Bureau or any other sponsor.
Code can be found in our GitHub repository: https://github.com/anonymous-conf/dplr

the solution to the following optimization problem:

$$(\hat{\alpha}, \hat{\beta}) = \arg\min_{\alpha,\beta} \|\mathbf{y} - \alpha\mathbf{x} - \beta\mathbf{1}\|_2. \quad (1)$$

It is the most commonly used solution in practice since it is the maximum likelihood estimator when F_e is Gaussian, and it has a closed form solution. We define the following empirical quantities:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$
$$\mathrm{ncov}(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x} - \bar{x}\mathbf{1}, \mathbf{y} - \bar{y}\mathbf{1}\rangle,$$
$$\mathrm{nvar}(\mathbf{x}) = \langle \mathbf{x} - \bar{x}\mathbf{1}, \mathbf{x} - \bar{x}\mathbf{1}\rangle = n \cdot \mathrm{var}(\mathbf{x}).$$

Then

$$\hat{\alpha} = \frac{\mathrm{ncov}(\mathbf{x}, \mathbf{y})}{\mathrm{nvar}(\mathbf{x})} \text{ and } \hat{\beta} = \bar{y} - \hat{\alpha}\bar{x} \quad (2)$$

this paper, we focus on predicting the (mean of the) response variable y at a single value of the explanatory variable x. For $x_{new} \in [0, 1]$, the *prediction* at x_{new} is defined as:

$$p_{x_{new}} = \alpha x_{new} + \beta$$

Let $\widehat{p}_{x_{new}}$ be the estimate of the prediction at x_{new} computed using the OLS estimates $\hat{\alpha}$ and $\hat{\beta}$. The quantity $\widehat{p}_{x_{new}}$ is a random variable, where the randomness is due to the sampling process. The standard error $\widehat{\sigma}(\widehat{p}_{x_{new}})$ is an estimate of the standard deviation of $\widehat{p}_{x_{new}}$. If we assume that the noise F_e is Gaussian, then we can compute the standard error as follows (see [26], for example):

$$\widehat{\sigma}(\widehat{p}_{x_{new}}) = \frac{\|\mathbf{y} - \hat{\alpha}\mathbf{x} - \hat{\beta}\|_2}{\sqrt{n-2}}\sqrt{\frac{1}{n} + \frac{(x_{new} - \bar{x})^2}{n\mathrm{var}(\mathbf{x})}}. \quad (3)$$

It can be shown that the variance of $(\widehat{p}_{x_{new}} - p_{x_{new}})/\widehat{\sigma}(\widehat{p}_{x_{new}})$ approaches 1 as n increases.

1.1.2 Differential Privacy. In this section, we define the notion of privacy that we are using: differential privacy (DP). Since our algorithms often include hyperparameters, we include a definition of DP for algorithms that take as input not only the dataset, but also the desired privacy parameters and any required hyperparameters. Let $\mathcal{X}$ be a data universe and $\mathcal{X}^n$ be the space of datasets. Two datasets $d, d' \in \mathcal{X}^n$ are neighboring, denoted $d \sim d'$, if they differ on a single record. Let $\mathcal{H}$ be a hyperparameter space and $\mathcal{Y}$ be an output space.

Definition 1 ((ε, δ)-Differential Privacy [17]). *A randomized mechanism $M : \mathcal{X}^n \times \mathbb{R}_{\geq 0} \times [0, 1] \times \mathcal{H} \to \mathcal{Y}$ is* differentially private *if for all datasets $d \sim d' \in \mathcal{X}^n$, privacy-loss parameters $\varepsilon \geq 0, \delta \in [0, 1]$, hyperparams $\in \mathcal{H}$, and events E,*

$$\Pr[M(d, \varepsilon, \delta, hyperparams) \in E]$$
$$\leq e^{\varepsilon} \cdot \Pr[M(d', \varepsilon, \delta, hyperparams) \in E] + \delta,$$

where the probabilities are taken over the random coins of M.

For strong privacy guarantees, the privacy-loss parameter is typically taken to be a small constant less than 1 (note that $e^{\varepsilon} \approx 1+\varepsilon$ as $\varepsilon \to 0$), but we will sometimes consider larger constants such as $\varepsilon = 8$ to match our motivating application (described in Section 1.2).

The key intuition for this definition is that the distribution of outputs on input dataset d is almost indistinguishable from the distribution on outputs on input dataset d'. Therefore, given the output of a differentially private mechanism, it is impossible to confidently determine whether the input dataset was d or d'.

1.1.3 Other Notation. The DP estimates will be denoted by $\widetilde{p}_{x_{new}}$ and the OLS estimates by $\widehat{p}_{x_{new}}$. We will focus on data with values bounded between 0 and 1, so $0 \leq x_i, y_i \leq 1$ for $i = 1, \ldots, n$.

We will be primarily concerned with the predicted values at $x_{new} = 0.25$ and 0.75, which for ease of notation we denote as p_{25} and p_{75}, respectively. Correspondingly, we will use $\widehat{p}_{25}, \widehat{p}_{75}$ to denote the OLS estimates of the predicted values and $\widetilde{p}_{25}, \widetilde{p}_{75}$ to denote the DP estimates. Our use of the 25th and 75th percentiles is motivated by the Opportunity Atlas tool [11], described in Section 1.2, which releases estimates of p_{25} and p_{75} for certain regressions done for every census tract in each state.

1.1.4 Error Metric. We will be concerned primarily with empirical performance measures. In particular, we will restrict our focus to high probability error bounds that can be accurately computed empirically through Monte Carlo experiments. The question of providing tight theoretical error bounds for DP linear regression is an important one, which we leave to future work. Since the relationship between the OLS estimate $\widehat{p}_{x_{new}}$ and the true value $p_{x_{new}}$ is well-understood, we focus on measuring the difference between the private estimate $\widetilde{p}_{x_{new}}$ and $\widehat{p}_{x_{new}}$, which is something we can measure experimentally on real datasets. Specifically, we define the *prediction error* at x_{new} to be $|\widetilde{p}_{x_{new}} - \widehat{p}_{x_{new}}|$.

For a dataset d, $x_{new} \in [0, 1]$, and $q \in [0, 100]$, we define the *q% error bound* as

$$C(q)(d) = \min\left\{c : \mathbb{P}(|\widetilde{p}_{x_{new}} - \widehat{p}_{x_{new}}| \leq c) \geq \frac{q}{100}\right\},$$

where the dataset d is fixed, and the probability is taken over the randomness in the DP algorithm.

We empirically estimate $C(q)$ by running many trials of the algorithm on the same dataset d:

$$\hat{C}(q)(d) = \min\left\{c : \text{for at least } q\% \text{ of trials}, |\widetilde{p}_{x_{new}} - \widehat{p}_{x_{new}}| \leq c\right\}.$$

We term $\hat{C}(q)(d)$ the *q% empirical error bound.* We will often drop the reference to d from the notation. This error metric only accounts for the randomness in the algorithm, not the sampling error.

When the ground truth is known (eg. for synthetically generated data), we can compute error bounds compared to the ground truth, rather than to the non-private OLS estimate. So, let $C_{\mathrm{true}}(q)(d)$ and $\hat{C}_{\mathrm{true}}(q)(d)$ be similar to the error bounds described earlier, except that the prediction error is measured as $|\widetilde{p}_{x_{new}} - p_{x_{new}}|$. This error metric accounts for the randomness in both the sampling and the algorithm. When we say *the noise added for privacy is less than the sampling error*, we are referring to the technical statement that $\hat{C}(68)$ is less than the standard error, $\widehat{\sigma}(\widehat{p}_{x_{new}})$.

1.2 Motivating Use-Case: Opportunity Atlas

The Opportunity Atlas, designed and deployed by the economics research group Opportunity Insights, is an interactive tool designed to study the link between the neighbourhood a child grows up in and their prospect for economic mobility [11]. The tool provides valuable insights to researchers and policy-makers, and is built from Census data, protected under Title 13 and authorized by the Census Bureau's Disclosure Review Board, linked with federal income tax returns from the US Internal Revenue Service. The Atlas provides individual statistics on each census tract in the country, with tract data often being refined by demographics to contain only a small subset of the individuals who live in that tract; for example, black male children with low parental income. The resulting datasets typically contain 100 to 400 datapoints, but can be as small as 30 datapoints. A separate regression estimate is released in each of these small geographic areas. The response variable $y_i \in [0,1]$ is the child's income percentile at age 35 and the explanatory variable $x_i \in [0,1]$ is the parent's income percentile, each with respect to the national income distribution. The coefficient α in the model $y_i = \alpha \cdot x_i + \beta + e_i$ is a measure of economic mobility for that particular Census tract and demographic. The small size of the datasets used in the Opportunity Atlas are the result of Chetty et al.'s motivation to study inequality at the neighbourhood level: "the estimates permit precise targeting of policies to improve economic opportunity by uncovering specific neighborhoods where certain subgroups of children grow up to have poor outcomes. Neighborhoods matter at a very granular level: conditional on characteristics such as poverty rates in a child's own Census tract, characteristics of tracts that are one mile away have little predictive power for a child's outcomes" [11].

1.3 Robustness and DP Algorithm Design

Simple linear regression is one of the most fundamental statistical tasks with well-understood convergence properties in the non-private literature. However, finding a differentially private estimator for this task that is accurate across a range of datasets and parameter regimes is surprisingly nuanced. As a first attempt, one might consider the global sensitivity [17]:

Definition 2 (Global Sensitivity)**.** *For a query $q : \mathcal{X}^n \to \mathbb{R}^k$, the **global sensitivity** is*

$$GS_q = \max_{\boldsymbol{x} \sim \boldsymbol{x}'} \|q(\boldsymbol{x}) - q(\boldsymbol{x}')\|_1.$$

One can create a differentially private mechanism by adding noise proportional to GS_q/ε. Unfortunately, the global sensitivity of p_{25} and p_{75} are both infinite (even though we consider bounded $\mathbf{x}, \mathbf{y} \in [0,1]^n$, the point estimates p_{25} and p_{75} are unbounded). For the type of datasets that we typically see in practice, however, changing one datapoint does not result in a major change in the point estimates. For such datasets, where the point estimates are reasonably stable, one might hope to take advantage of the local sensitivity:

Definition 3 (Local Sensitivity [21])**.** *The **local sensitivity** of a query $q : \mathcal{X}^n \to \mathbb{R}^k$ with respect to a dataset $\boldsymbol{x}$ is*

$$LS_q(\boldsymbol{x}) = \max_{\boldsymbol{x} \sim \boldsymbol{x}'} \|q(\boldsymbol{x}) - q(\boldsymbol{x}')\|_1.$$

Adding noise proportional to the local sensitivity is typically **not** differentially private, since the local sensitivity itself can reveal information about the underlying dataset. To get around this, one could try to add noise that is larger, but not too much larger, than the local sensitivity. As DP requires that the amount of noise added cannot depend too strongly on the particular dataset, DP mechanisms of this flavor often involve calculating the local sensitivity on neighbouring datasets. So far, we have been unable to design a computationally feasible algorithm for performing the necessary computations for OLS. Furthermore, computationally feasible upper bounds for the local sensitivity have, so far, proved too loose to be useful.

The Opportunity Insights (OI) algorithm takes a heuristic approach by adding noise proportional to a non-private, heuristic upper bound on the local sensitivity of data from tracts in any given state. However, their heuristic approach does not satisfy the formal requirements of differential privacy, leaving open the possibility that there is a realistic attack.

The OI algorithm incorporates a "winsorization" step in their estimation procedure (e.g. dropping bottom and top 10% of data values). This sometimes has the effect of greatly reducing the local sensitivity (and also their upper bound on it) due to the possible removal of outliers. This suggests that for finding an effective differentially private algorithm, we should consider differentially private analogues of *robust* linear regression methods rather than of OLS. Specifically, we consider *Theil-Sen*, a robust estimator for linear regression proposed by Theil [24] and further developed by Sen [22]. Similar to the way in which the median is less sensitive to changes in the data than the mean, the Theil-Sen estimator is more robust to changes in the data than OLS .

In this work, we consider three differentially private algorithms based on both robust and non-robust methods:

- **`NoisyStats`** is the DP mechanism that most closely mirrors OLS. It involves perturbing the sufficient statistics $\text{ncov}(\mathbf{x}, \mathbf{y})$ and $\text{nvar}(\mathbf{x})$ from the OLS computation. This algorithm is inspired by the "Analyze Gauss" technique [18], although the noise we add ensures pure differential privacy rather than approximate differential privacy. `NoisyStats` has two main benefits: it is as computationally efficient as its non-private analogue, and it allows us to release DP versions of the sufficient statistics with no extra privacy cost.
- **`DPTheilSen`** is a DP version of Theil-Sen. The non-private estimator computes the p_{25} estimates based on the lines defined by all pairs of datapoints $(x_i, y_i), (x_j, y_j) \in [0,1]^2$ for all $i \neq j \in [n]$, then outputs the median of these pairwise estimates. To create a differentially private version, we replace the median computation with a differentially private median algorithm. We will consider three DP versions of this algorithm which use different DP median algorithms: `DPExpTheilSen`, `DPWideTheilSen`, and `DPSSTheilSen`. We also consider more computationally efficient variants that pair points according to one or more random matchings, rather than using all $\binom{n}{2}$ pairs. A DP algorithm obtained by using one matching was previously considered by Dwork and Lei [16] (their "Short-Cut Regression Algorithm"). Our algorithms can be viewed as updated versions, reflecting improvements in DP median estimation since [16], as well as incorporating benefits accrued by considering more than one matching or all $\binom{n}{2}$ pairs.
- **`DPGradDescent`** is a DP mechanism that uses DP gradient descent to solve the convex optimization problem that defines OLS: $\text{argmin}_{\alpha,\beta} \|\mathbf{y} - \alpha\mathbf{x} - \beta\mathbf{1}\|_2$. We use the private stochastic gradient descent technique proposed by Bassily

et. al in [4]. Versions that satisfy pure, approximate, and zero-concentrated differential privacy are considered.

1.4 Our Results

Our experiments indicate that for a wide range of realistic datasets, and moderate values of ε, it is possible to choose a DP linear regression algorithm where the error due to privacy is less than the standard error. In particular, in our motivating use-case of the Opportunity Atlas, we can design a differentially private algorithm that outperforms the heuristic method used by the Opportunity Insights team. This is promising, since the error added by the heuristic method was deemed acceptable by the Opportunity Insights team for deployment of the tool, and for use by policy makers. One particular differentially private algorithm of the robust variety, called `DPExpTheilSen`, emerges as the best algorithm in a wide variety of settings for this small-dataset regime.

Our experiments reveal three main settings where an analyst should consider alternate algorithms:

- When $\varepsilon n\text{var}(\mathbf{x})$ is large and σ_e^2 is large, a DP algorithm `NoisyStats` that simply perturbs the Ordinary Least Squares (OLS) sufficient statistics, $\text{nvar}(\mathbf{x})$ and $\text{ncov}(\mathbf{x}, \mathbf{y})$, performs well. This algorithm is preferable in this setting since it is more computationally efficient, and allows for the release of noisy sufficient statistics without any additional privacy loss.
- When the standard error $\widehat{\sigma}(\widehat{p}_{x_{new}})$ is very small, `DPExpTheilSen` can perform poorly. In this setting, one should switch to a different DP estimator based on Theil-Sen. We give two potential alternatives, which are both useful in different situations.
- The algorithm `DPExpTheilSen` requires as input a range in which to search for the output predicted value. If this range is very large and ε is small ($\varepsilon \ll 1$) then `DPExpTheilSen` can perform poorly, so it is good to use other algorithms like `NoisyStats` that do not require a range for the output (but do require that the input variables x_i and y_i are bounded, which is not required for `DPExpTheilSen`).

The quantity $\varepsilon n\text{var}(\mathbf{x})$ is connected to the size of the dataset, how concentrated the independent variable of the data is and how private the mechanism is. Experimentally, this quantity has proved to be a good indicator of the relative performance of the DP algorithms. Roughly, when $\varepsilon n\text{var}(\mathbf{x})$ is small, the OLS estimate can be very sensitive to small changes in the data, and thus we recommend switching to differentially private versions of the Theil-Sen estimator. In the opposite regime, when $\varepsilon n\text{var}(\mathbf{x})$ is large, `NoisyStats` typically suffices and is a simple non-robust method to adopt in practice. In this regime the additional noise added for privacy by `NoisyStats` can be less than the difference between the non-private OLS and Theil-Sen point estimates. The other non-robust algorithm we consider, `DPGradDescent`, may be a suitable replacement for `NoisyStats` depending on the privacy model used (i.e. pure, approximate, or zero-concentrated DP). Our comparison of `NoisyStats` and `DPGradDescent` is not comprehensive, but we find that any performance advantages of `DPGradDescent` over `NoisyStats` appear to be small in the regime where the non-robust methods outperform the robust ones. `NoisyStats` is simpler and has fewer hyperparameters, however, so we find that it may be preferable in practice.

In addition to the quantity $\varepsilon n\text{var}(\mathbf{x})$, the magnitude of noise in the dependent variable effects the relative performance of the algorithms. When the dependent variable is not very noisy (i.e. σ_e^2 is small), the Theil-Sen-based estimators perform better since they are better at leveraging a strong linear signal in the data.

These results show that even in this one-dimensional setting, the story is already quite nuanced. Indeed, which algorithm performs best depends on properties of the dataset, such as $n\text{var}(\mathbf{x})$, which cannot be directly used without violating differential privacy. So, one has to make the choice based on guesses (eg. using similar public datasets) or develop differentially private methods for selecting the algorithm, a problem which we leave to future work. Moreover, most of our methods come with hyperparameters that govern their behavior. How to optimally choose these parameters is still an open problem. In addition, we focus on outputting accurate point estimates, rather than confidence intervals. Computing confidence intervals is an important direction for future work.

1.5 Other Related Work

Linear regression is one of the most prevalent statistical methods in the social sciences, and hence has been studied previously in the differential privacy literature. These works have included both theoretical analysis and experimental exploration, with the majority of work focusing on large datasets.

One of our main findings — that robust estimators perform better than parametric estimators in the differentially private setting, even when the data come from a parametric model — corroborate insights by [16] with regard to the connection between robust statistics and differential privacy, and by [14] in the context of hypothesis testing.

Other systematic studies of DP linear regression have been performed by Sheffet [23] and Wang [25]. Sheffet [23] considered differentially private ordinary least squares methods and estimated confidence intervals for the regression coefficients. He assumes normality of the explanatory variables, while we do not make any distributional assumptions on our covariates.

Private linear regression in the high-dimensional settings is studied by Cai et al. [8] and Wang [25]. Wang [25] considered private ridge regression, where the ridge parameter is adaptively and privately chosen, using techniques similar to output perturbation [9]. These papers present methods and experiments for high dimensional data (the datasets used contain at least 13 explanatory variables), whereas we are concerned with the one-dimensional setting. We find that even the one-dimensional setting the choice of optimal algorithm is already quite complex.

A Bayesian approach to DP linear regression is taken by Bernstein and Sheldon [5]. Their method uses sufficient statistic perturbation (similar to our `NoisyStats` algorithm) for private release and Markov Chain Monte Carlo sampling over a posterior of the regression coefficients. Their Bayesian approach can produce tight credible intervals for the regression coefficients, but unlike ours, requires distributional assumptions on both the regression coefficients and the underlying independent variables. In order to make private releases, we assume the data is bounded but make no other distributional assumptions on the independent variables.

2 ALGORITHMS

In this section we detail the practical differentially private algorithms we will evaluate experimentally. Pseudocode for all efficient implementations of each algorithm described can be found in the Appendix, and real code can be found in our GitHub repository. We will assume throughout that $(x_i, y_i) \in [0,1]^2$, as in the Opportunity Atlas, where it is achieved by preprocessing of the data. While we would ideally ensure differentially private preprocessing of the data, we will consider this outside the scope of this work.

2.1 `NoisyStats`

In `NoisyStats` (Algorithm 1), we add Laplace noise, with standard deviation approximately $1/\varepsilon$, to the OLS sufficient statistics, $\text{ncov}(\mathbf{x}, \mathbf{y})$, $\text{nvar}(\mathbf{x})$, and then use the noisy sufficient statistics to compute the predicted values. Note that this algorithm fails if the denominator for the OLS estimator, the noisy version of $\text{nvar}(\mathbf{x})$, becomes 0 or negative, in which case we output $\perp$ (failure). The probability of failure decreases as ε or $\text{nvar}(x)$ increases. `NoisyStats` is the simplest and most efficient algorithm that we will study. In addition, the privacy guarantee is maintained even if we additionally release the noisy statistics $\text{nvar}(\mathbf{x}) + L_1$ and $\text{ncov}(\mathbf{x}, \mathbf{y}) + L_2$, which may be of independent interest to researchers. We also note that the algorithm is biased due to dividing by a Laplacian distribution centered at $\text{nvar}(\mathbf{x})$.

Algorithm 1: `NoisyStats`: $(\varepsilon, 0)$-DP Algorithm

```
Data: $\{(x_i, y_i)\}_{i=1}^n \in ([0,1] \times [0,1])^n$
Privacy params: $\varepsilon$
Hyperparams: n/a
Define $\Delta_1 = \Delta_2 = (1 - 1/n)$
Sample $L_1 \sim \text{Lap}(0, 3\Delta_1/\varepsilon)$
Sample $L_2 \sim \text{Lap}(0, 3\Delta_2/\varepsilon)$
if $nvar(\mathbf{x}) + L_2 > 0$ then
    $\tilde{\alpha} = \frac{\text{ncov}(\mathbf{x},\mathbf{y})+L_1}{\text{nvar}(\mathbf{x})+L_2}$
    $\Delta_3 = 1/n \cdot (1 + |\tilde{\alpha}|)$
    Sample $L_3 \sim \text{Lap}(0, 3\Delta_3/\varepsilon)$
    $\tilde{\beta} = (\bar{y} - \tilde{\alpha}\bar{x}) + L_3$
    $\widetilde{p}_{25} = 0.25 \cdot \tilde{\alpha} + \tilde{\beta}$
    $\widetilde{p}_{75} = 0.75 \cdot \tilde{\alpha} + \tilde{\beta}$
    return $\widetilde{p}_{25}, \widetilde{p}_{75}$
else
    return $\perp$
```

Lemma 4. *Algorithm 1 (*`NoisyStats`*) is $(\varepsilon, 0)$-DP.*

2.2 DP TheilSen

The standard Theil-Sen estimator is a *robust* estimator for linear regression. It computes the p_{25} estimates based on the lines defined by all pairs of datapoints $(x_i, y_i), (x_j, y_j) \in [0,1]^2$ for all $i \neq j \in [n]$, then outputs the median of these pairwise estimates. To create a differentially private version, we can replace the median computation with a differentially private median algorithm. We implement this approach using three DP median algorithms; two based on the exponential mechanism [20] and one based on the smooth sensitivity of [21] and the noise distributions of [7].

In the "complete" version of Theil-Sen, all pairwise estimates are included in the final median computation. A similar algorithm can be run on the point estimates computed using k random matchings of the (x_i, y_i) pairs. The case $k = 1$ amounts to the differentially private "Short-cut Regression Algorithm" proposed by Dwork and Lei [16]. This results in a more computationally efficient algorithm.

Furthermore, while in the non-private setting $k = n - 1$ is the optimal choice, this is not necessarily true when we incorporate privacy. Each datapoint affects k of the datapoints in $\mathbf{z}^{(p25)}$ (using the notation from Algorithm 2) out of the total $k \cdot n/2$ datapoints. In some private algorithms, the amount of noise added for privacy is a function of the fraction of points in $\mathbf{z}^{(p25)}$ influenced by each (x_i, y_i), which is independent of k – meaning that one should always choose k as large as is computationally feasible. However, this is not always the case. Increasing k can result in more noise being added for privacy resulting in a trade-off between decreasing the noise added for privacy (with smaller k) and decreasing the non-private error (with larger k).

While intuitively, for small k, one can think of using k random matchings, we will actually ensure that no edge is included twice. We say the permutations $\tau_1, \cdots, \tau_{n-1}$ are a decomposition of the complete graph on n vertices, K_n into $n-1$ matchings if $\cup \Sigma_k = \{(x_i, x_j) \mid i, j \in [n]\}$ where $\Sigma_k = \{(x_{\tau_k(1)}, x_{\tau_k(2)}), \cdots, (x_{\tau_k(n-1)}, x_{\tau_k(n)})\}$. Thus, `DPTheilSen(n-1)Match`, referred to simply as `DPTheilSen`, uses all pairs of points. We will focus mainly on $k = n-1$, which we will refer to simply as `DPTheilSen` and $k = 1$, which we will refer to as `DPTheilSenMatch`. For any other k, we denote the algorithm as `DPTheilSenkMatch`. In the following subsections we discuss the different differentially private median algorithms we use as subroutines. The pseudo-code for `DPTheilSenkMatch` is found in Algorithm 2.

Lemma 5. *If $DPmed(\mathbf{z}^{(p25)}, \varepsilon, (n, k, hyperparameters) = \mathcal{M}(z^{(p25)}, hyperparameters))$ for some $(\varepsilon/k, 0)$-DP mechanism $\mathcal{M}$, then Algorithm 2 (*`DPTheilSenkMatch`*) is $(\varepsilon, 0)$-DP.*

2.2.1 *DP Median using the Exponential Mechanism.*

The first differentially private algorithm for the median that we will consider is an instantiation of the exponential mechanism [20], a differentially private algorithm designed for general optimization problems. The exponential mechanism is defined with respect to a utility function u, which maps (dataset, output) pairs to real values. For a dataset $\mathbf{z}$, the mechanism aims to output a value r that maximizes $u(\mathbf{z}, r)$.

Definition 6 (Exponential Mechanism [20]). *Given dataset $\mathbf{z} \in \mathbb{R}^n$ and the range of the outputs, $[r_l, r_u]$, the exponential mechanism outputs $r \in [r_l, r_u]$ with probability proportional to $\exp\left(\frac{\varepsilon u(\mathbf{z}, r)}{2GS_u}\right)$, where*

$$GS_u = \max_{r \in [r_l, r_u]} \max_{\mathbf{z}, \mathbf{z}' \text{ neighbors}} |u(\mathbf{z}, r) - u(\mathbf{z}', r)|.$$

One way to instantiate the exponential mechanism to compute the median is by using the following utility function. Let

$$u(\mathbf{z}, r) = -\left|\#\text{above } r - \#\text{below } r\right|$$

Algorithm 2: DPTheilSenkMatch: $(\varepsilon, 0)$-DP Algorithm

```
Data: $\{(x_i, y_i)\}_{i=1}^n \in ([0,1] \times [0,1])^n$
Privacy params: $\varepsilon$
Hyperparams: $n, k$, DPmed, hyperparams
$\mathbf{z}^{(p25)}, \mathbf{z}^{(p75)} = [\ ]$
Let $\tau_1, \cdots, \tau_{n-1}$ be a decomposition of $K_n$ into matchings.
for $k$ iterations do
    Sample (without replacement) $h \in [n-1]$
    for $0 \le i < n-1, i = i+2$ do
        $j = \tau_h(i)$
        $l = \tau_h(i+1)$
        if $(x_l - x_j \neq 0)$ then
            $s = (y_l - y_j)/(x_l - x_j)$
            $z_{j,l}^{(p25)} = s\left(0.25 - \frac{x_l + x_j}{2}\right) + \frac{y_l + y_j}{2}$
            $z_{j,l}^{(p75)} = s\left(0.75 - \frac{x_l + x_j}{2}\right) + \frac{y_l + y_j}{2}$
            Append $z_{j,l}^{(p25)}$ to $\mathbf{z}^{(p25)}$ and $z_{j,l}^{(p75)}$ to $\mathbf{z}^{(p75)}$
$\tilde{p}_{25} = \text{DPmed}\left(\mathbf{z}^{(p25)}, \varepsilon, (n, k, \text{hyperparams})\right)$
$\tilde{p}_{75} = \text{DPmed}\left(\mathbf{z}^{(p75)}, \varepsilon, (n, k, \text{hyperparams})\right)$
return $\tilde{p}_{25}, \tilde{p}_{75}$
```

where #above r and #below r denote the number of datapoints in $\mathbf{z}$ that are above and below r in value respectively, not including r itself. An example of the shape of the output distribution of this algorithm is given in Figure 1. An efficient implementation is given in the Appendix.

We will write `DPExpTheilSenkMatch` to refer to `DPTheilSenkMatch` where DPmed is the DP exponential mechanism described above with privacy parameter ε/k. Again, we write `DPExpTheilSenMatch` when $k = 1$ and `DPExpTheilSen` when $k = n-1$.

Lemma 7. `DPExpTheilSenkMatch` *is* $(\varepsilon, 0)$*-DP.*

2.2.2 *DP Median using Widened Exponential Mechanism.*

When the output space is the real line, the standard exponential mechanism for the median has some nuanced behaviour when the data is highly concentrated. For example, imagine in Figure 1 if all the datapoints coincided. In this instance, `DPExpTheilSen` is simply the uniform distribution on $[r_l, r_u]$, despite the fact that the median of the dataset is very stable. To mitigate this issue, we use a variation on the standard utility function. For a widening parameter $\theta > 0$, the widened utility function is

$$u(\mathbf{z}, r) = -\min\{|\#\text{above } a - \#\text{below } a| \ : \ |a - r| \le \theta\}$$

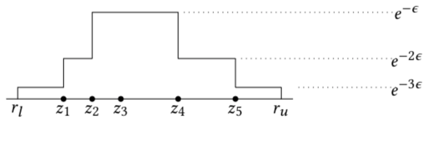


Figure 1: Unnormalized distribution of outputs of the exponential mechanism for differentially privately computing the median of dataset z.

where #above a and #below a are defined as before. This has the effect of increasing the probability mass around the median, as shown in Figure 2.

The parameter θ needs to be carefully chosen. All outputs within θ of the median are given the same utility score, so θ represents a lower bound on the error. Conversely, choosing θ too small may result in the area around the median not being given sufficient weight in the sampled distribution. Note that $\theta > 0$ is only required when one expects the datasets $\mathbf{z}^{p25}$ to be highly concentrated (for example when the dataset has strong linear signal). We defer the question of optimally choosing θ to future work.

An efficient implementation of the θ-widened exponential mechanism for the median can be found in the Appendix as Algorithm 4. We will use `DPWideTheilSenkMatch` to refer to `DPTheilSenkMatch` where DPmed is the θ-widened exponential mechanism with privacy parameter ε/k. Again, we use `DPWideTheilSenMatch` when $k = 1$ and `DPWideTheilSen` when $k = n-1$.

Lemma 8. `DPWideTheilSenkMatch` *is* $(\varepsilon, 0)$*-DP.*

2.2.3 *DP Median using Smooth Sensitivity Noise Addition.*

The final algorithm we consider for releasing a differentially private median adds noise scaled to the smooth sensitivity – a smooth upper bound on the local sensitivity function. Intuitively, this algorithm should perform well when the datapoints are clustered around the median; that is, when the median is very stable.

Definition 9 (Smooth Upper Bound on LS_f [21])**.** *For* $t > 0$*, a function* $S_{f,t} : \mathcal{X}^n \to \mathbb{R}$ *is a* t*-smooth upper bound on the local sensitivity of a function* $f : \mathcal{X}^n \to \mathbb{R}$ *if:*

$$\forall \mathbf{z} \in \mathcal{X}^n : LS_f(\mathbf{z}) \le S_{f,t}(\mathbf{z});$$

$$\forall \mathbf{z}, \mathbf{z}' \in \mathcal{X}^n, d(\mathbf{z}, \mathbf{z}') = 1 : S_{f,t}(\mathbf{z}) \le e^t \cdot S_{f,t}(\mathbf{z}').$$

where $d(\mathbf{z}, \mathbf{z}')$ *is the distance between datasets* $\mathbf{z}$ *and* $\mathbf{z}'$*.*

Let $\mathcal{Z}_k : ([0,1] \times [0,1])^n \to \mathbb{R}^{kn/2}$ to denote the function that transforms a set of point coordinates into estimates for each pair of points in our k matchings, so in the notation of Algorithm 2, $\mathbf{z}^{p25} = \mathcal{Z}(\mathbf{x}, \mathbf{y})$. The function that we are concerned with the smooth sensitivity of is med$\circ\mathcal{Z}_k$, which maps $(\mathbf{x}, \mathbf{y})$ to med$(\mathbf{z}^{p25})$. We will use the following smooth upper bound to the local sensitivity:

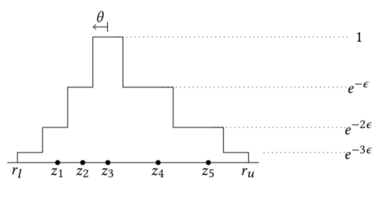


Figure 2: Unnormalized distribution of outputs of the θ-widened exponential mechanism for differentially privately computing the median of dataset z.

Lemma 10. *Let $z_1 \leq z_2 \leq \cdots \leq z_{2m}$ be a sorting of $\mathcal{Z}_k(\boldsymbol{x}, \boldsymbol{y})$. Then*

$$S^k_{med\circ\mathcal{Z},t}((\boldsymbol{x}, \boldsymbol{y})) = \max\Big\{z_{m+k} - z_m, z_m - z_{m-k}, \max_{l=1,\ldots,n}\max_{s=0,\cdots,k(l+1)} e^{-lt}(z_{m+s} - z_{m-(k(l+1)+s)})\Big\},$$

is a t-smooth upper bound on the local sensitivity of med$\circ\mathcal{Z}_k$.

Proof. Proof in the Appendix. □

The algorithm then adds noise proportional to $S^k_{\text{med}\circ\mathcal{Z},t}((\mathbf{x}, \mathbf{y}))/\varepsilon$ to med$\circ\mathcal{Z}(\mathbf{x}, \mathbf{y})$. A further discussion of the formula $S^k_{\text{med}\circ\mathcal{Z},t}((\mathbf{x}, \mathbf{y}))$ and pesudo-code can be found in Appendix E. The noise is sampled from the Student's T distribution. There are several other valid choices of noise distributions (see [21] and [7]), but we found the Student's T distribution to be preferable as the mechanism remains stable across values of ε

Lemma 11. `DPSSTheilSenkMatch` *is $(\varepsilon, 0)$-DP.*

2.3 DP Gradient Descent

Ordinary Least Squares (OLS) for simple 1-dimensional linear regression is defined as the solution to the optimization problem

$$\underset{\alpha,\beta}{\text{argmin}} \sum_{i=1}^n (y_i - \alpha x_i - \beta)^2.$$

There has been an extensive line of work on solving convex optimization problems in a differentially private manner. We use the private gradient descent algorithm of [4] to provide private estimates of the 0.25, 0.75 predictions (p_{25}, p_{75}). This algorithm performs standard gradient descent, except that noise is added to a clipped version of the gradient at each round (clipped to range $[-\tau, \tau]^2$ for some setting of $\tau > 0$). The number of calls to the gradient needs to be chosen carefully since we have to split our privacy budget amongst the gradient calls. We note that we are yet to fully explore the full suite of parameter settings for this method. See the Appendix for pseudo-code for our implementation and [4] for more information on differentially private stochastic gradient descent. We consider three versions of `DPGradDescent`: `DPGDPure`, `DPGDApprox`, and `DPGDzCDP`, which are $(\varepsilon, 0)$-DP, (ε, δ)-DP, and $(\varepsilon^2/2)$-zCDP algorithms, respectively. Zero-concentrated differential privacy (zCDP) is a slightly weaker notion of differential privacy that is well suited to iterative algorithms. For the purposes of this paper, we set $\delta = 2^{-30}$ whenever `DPGDApprox` is run, and set the privacy parameter of `DPGDzCDP` to allow for a fair comparison. `DPGDPure` provides the strongest privacy guarantee followed by `DPGDApprox` and then `DPGDzCDP`. As expected, `DPGDzCDP` typically has the best performance.

2.4 NoisyIntercept

We also compare the above DP mechanisms to simply adding noise to the average y-value. For any given dataset $(\mathbf{x}, \mathbf{y}) \in ([0, 1] \times [0, 1])^n$, this method computes $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$ and outputs a noisy estimate $\tilde{y} = \bar{y} + \text{Lap}\left(0, \frac{1}{\varepsilon n}\right)$ as the predicted $\widetilde{p}_{25}, \widetilde{p}_{75}$ estimates. This method performs well when the slope α is very small.

2.5 A Note on Hyperparameters

We leave the question of how to choose the optimal hyperparameters for each algorithm to future work. Unfortunately, since the optimal hyperparameter settings may reveal sensitive information about the dataset, one can not tune the hyperparameters on a hold-out set. However, we found that for most of the hyperparameters, once a good choice of hyperparameter setting was found, it could be used for a variety of similar datasets. Thus, one realistic way to tune the parameters may be to tune on a public dataset similar to the dataset of interest. For example, for applications using census data, one could tune the parameters on previous years' census data.

We note that, as presented here, `NoisyStats` requires no hyperparameter tuning, while `DPExpTheilSen` and `DPSSTheilSen` only require a small amount of tuning (they require upper and lower bounds on the output). However, `NoisyStats` requires knowledge of the range of the inputs x_i, y_i, which is not required by the Theil-Sen methods. Both `DPGradDescent` and `DPWideTheilSen` can be quite sensitive to the choice of hyperparameters. In fact, `DPExpTheilSen` is simply `DPWideTheilSen` with $\theta = 0$, and we will often see a significant difference in the behaviour of these algorithms. Preliminary experiments on the robustness of `DPWideTheilSen` to the choice of θ can be found in the Appendix.

3 EXPERIMENTAL OUTLINE

The goal of this work is to provide insight and guidance into what features of a dataset should be considered when choosing a differentially private algorithm for simple linear regression. As such, in the following sections, we explore the behavior of these algorithms on a variety of real datasets. We also consider some synthetic datasets where we can further explore how properties of the dataset affect performance.

3.1 Description of the Data

3.1.1 Opportunity Insights data. The first dataset we consider is a simulated version of the data used by the Opportunity Insights team in creating the Opportunity Atlas tool described in Section 1.2. In the appendix, we describe the data generation process used by the Opportunity Insights team to generate the simulated data. Each datapoint, (x_i, y_i), corresponds to a pair, (parent income percentile rank, child income percentile rank) , where parent income is defined as the total pre-tax income at the household level, averaged over the years 1994-2000, and child income is defined as the total pre-tax income at the individual level, averaged over the years 2014-2015 when the children are between the ages of 31-37. In the Census data, every state in the United States is partitioned into small neighborhood-level blocks called tracts. We perform the linear regression on each tract individually. The "best" differentially private algorithm differs from state to state, and even tract to tract. We focus on results for Illinois (IL) which has a total of $n = 219,594$ datapoints divided among 3,108 tracts. The individual datasets (corresponding to tracts in the Census data) each contain between $n = 30$ and $n = 400$ datapoints. The statistic $n\text{var}(\mathbf{x})$ ranges between 0 and 25, with the majority of tracts having $n\text{var}(\mathbf{x}) \in [0, 5]$. We use $\varepsilon = 16$ in all our experiments on this data since that is the value approved by the Census for, and currently used in, the release of the Opportunity Atlas [11].

3.1.2 *Washington, DC Bikeshare UCI Dataset.*

Next, we consider a family of small datasets containing data from the Capital Bikeshare system, in Washington D.C., USA, from the years 2011 and 2012[1]. Each (x_i, y_i) datapoint of the dataset contains the temperature (x_i) and user count of the bikeshare program (y_i) for a single hour in 2011 or 2012. The x_i and y_i values are both normalized so that they lie between 0 and 1 by a linear rescaling of the data. In order to obtain smaller datasets to work with, we segment this dataset into 288 (12x24) smaller datasets each corresponding to a (month, hour of the day) pair. Each smaller datasetcontains between $n = 45$ and $n = 62$ datapoints. We linearly regress the number of active users of the bikeshare program on the temperature. While the variables are positively correlated, the correlation is not obviously linear — in that the data is not necessarily well fit by a line — within many of these datasets, so we see reasonably large standard error values ranging between 0.0003 and 0.32 for this data. Note that our privacy guarantee is at the level of hours rather than individuals so it serves more as an example to evaluate the utility of our methods, rather than a real privacy application. While this is an important distinction to make for deployment of differential privacy, we will not dwell on it here since our goal to evaluate the statistical utility of our algorithms on real datasets. An example of one of these datasets is displayed in Figure 3.

3.1.3 *Carbon Nanotubes UCI Dataset.*

While we are primarily concerned with evaluating the behavior of the private algorithms on small datasets, we also present results on a good candidate for private analysis: a large dataset with a strong linear relationship. In contrast to the Bikeshare UCI data and the Opportunity Insights data, this is a single dataset, rather than a family of datasets. This data is drawn from a dataset studying atomic coordinates in carbon nanotubes [3]. For each datapoint (x_i, y_i), x_i is the u-coordinate of the initial atomic coordinates of a molecule and y_i is the u-coordinate of the calculated atomic coordinates after the energy of the system has been minimized. After we filtered out all points that are not contained in $[0, 1] \times [0, 1]$, the resulting datasetcontains $n = 10,683$ datapoints. A graphical representation of the data is included in Figure 4. Due to the size of this dataset, we run the efficient `DPTheilSenMatch` algorithm with $k = 1$ on this dataset. This dataset does not contain sensitive information, however we have included it to evaluate the behaviour of the DP algorithms on a variety of real datasets.

3.1.4 *Stock Exchange UCI Dataset.*

Our final real dataset studies the relationship between the Istanbul Stock Exchange and the USD Stock Exchange [1]. [2] It compares $\{x_i$ = Istanbul Stock Exchange national 100 index$\}$ to $\{y_i$ = the USD International Securities Exchange$\}$. This dataset has $n = 250$ datapoints and a representation is included in Figure 5. This is a a smaller, noisier dataset than the Carbon Nanotubes UCI dataset.

3.1.5 *Synthetic Data.*

The synthetic datasets were constructed by sampling $x_i \in \mathbb{R}$, for $i = 1, \ldots, n$, independently from a uniform distribution with $\bar{x} = 0.5$ and variance σ_x^2. For each x_i, the corresponding y_i is generated as $y_i = \alpha x_i + \beta + e_i$, where $\alpha = 0.5$, $\beta = 0.2$, and e_i is sampled from $\mathcal{N}(0, \sigma_e^2)$. The (x_i, y_i) datapoints are then clipped to the box $[0, 1]^2$. The DP algorithms estimate the prediction at x_{new} using privacy parameter ε.

The values of n, σ_x^2, σ_e^2, x_{new}, and ε vary across the individual experiments. The synthetic data experiments are designed to study which properties of the data and privacy regime determine the performance of the private algorithms. Thus, in these experiments, we

[1]This data is publicly available at http://capitalbikeshare.com/system-data [19].

[2]This dataset was collected from imkb.gov.tr and finance.yahoo.com.

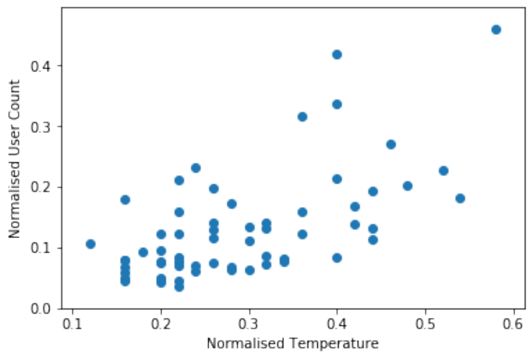


Figure 3: Example of dataset from Washington, DC Bikeshare UCI Dataset.

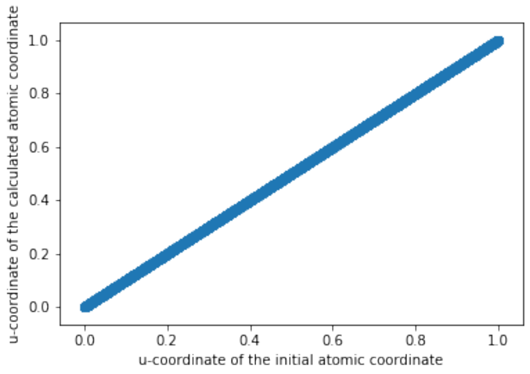


Figure 4: Carbon Nanotubes UCI Dataset.

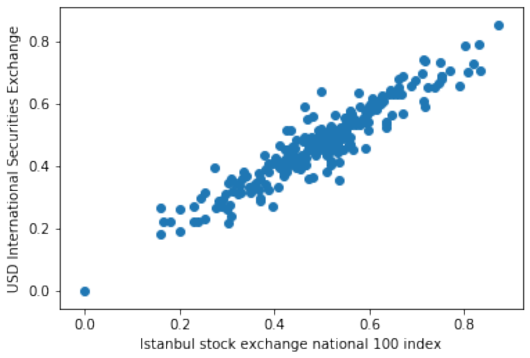


Figure 5: The Stock Exchange UCI Dataset.

vary one of parameters listed above and observe the impact on the accuracy of the algorithms and their relative performance to each other. Since we know the ground truth on this synthetically generated data, we plot empirical error bounds that take into account both the sampling error and the error due to the DP algorithms, $\hat{C}_{\text{true}}(68)/\sigma(\widehat{p}_{x_{new}})$. We evaluate `DPTheilSenkMatch` with $k = 10$ in the synthetic experiments rather than `DPTheilSen`, since the former is computationally more efficient and still gives us insight into the performance of the latter.

3.2 Hyperparameters

Hyperparameters were tuned on the semi-synthetic Opportunity Insights data by experimenting with many different choices, and choosing the best. The hyperparameters are listed in Table 1. We leave the question of optimizing hyperparameters in a privacy-preserving way to future work. Hyperparameters for the UCI and synthetic datasets were chosen according to choices that seemed to perform well on the Opportunity Insights datasets. This ensures that the hyperparameters were not tuned to the specific datasets (ensuring validity of the privacy guarantees), but also leaves some room for improvement in the performance by more careful hyperparameter tuning.

4 RESULTS AND DISCUSSION

In this section we discuss the findings from the experiments described in the previous section. In the majority of the real datasets tested, `DPExpTheilSen` is the best performing algorithm. We'll discuss some reasons why `DPExpTheilSen` performs well, as well as some regimes when other algorithms perform better.

A note on the statement of privacy parameters in the experiments. We will state the privacy budget used to compute the pair (p_{25}, p_{75}), however we will only show empirical error bounds for p_{25}. The empirical error bounds for p_{75} display similar phenomena. The algorithms `NoisyStats` and `DPGradDescent` inherently release both point estimates together so the privacy loss is the same whether we release the pair (p_{25}, p_{75}) or just p_{25}. However, `DPExpTheilSen`, `DPExpWideTheilSen`, `DPSSTheilSen` and the algorithm used by the Opportunity Insights team use half their budget to release p_{25} and half their budget to release p_{75}, separately.

Table 1: Hyperparameters used in experiments on OI data and UCI datasets.

Algorithms	OI/UCI data	Synthetic data
`NoisyStats`	None	None
`NoisyIntercept`	None	N/A
`DPExpTheilSen`	$r_l = -0.5, r_u = 1.5$	$r_l = -2, r_u = 2$
`DPWideTheilSen`	$r_l = -0.5, r_u = 1.5$, $\theta = 0.01$	$r_l = -2, r_u = 2$ $\theta = 0.01$
`DPSSTheilSen`	$r_l = -0.5, r_u = 1.5$, $d = 3$	$r_l = -2, r_u = 2$ $d = 3$
`DPGradDescent`	$\tau = 1$, T = 80	$\tau = 1$, T = 80

4.1 Real Data and the Opportunity Insights Application

Figure 6 shows the results of all the DP linear regression algorithms, as well as the mechanism used by the Opportunity Insights team, on the Opportunity Insights data for the states of Illinois and North Carolina. For each algorithm, we build an empirical cumulative density distribution of the empirical error bounds set over the tracts in that state. The vertical dotted line in Figures 6 intercepts each curve at the point where the *noise due to privacy exceeds the standard error*. The Opportunity Insights team used a heuristic method inspired by, but not satisfying, DP to deploy the Opportunity Altas [11]. Their privacy parameter of $\varepsilon = 16$ was selected by the Opportunity Insights team and a Census Disclosure Review Board by balancing the privacy and utility considerations. Figure 6 shows that there exist differentially private algorithms that are competitive with, and in many cases more accurate than, the algorithm currently deployed in the Opportunity Atlas.

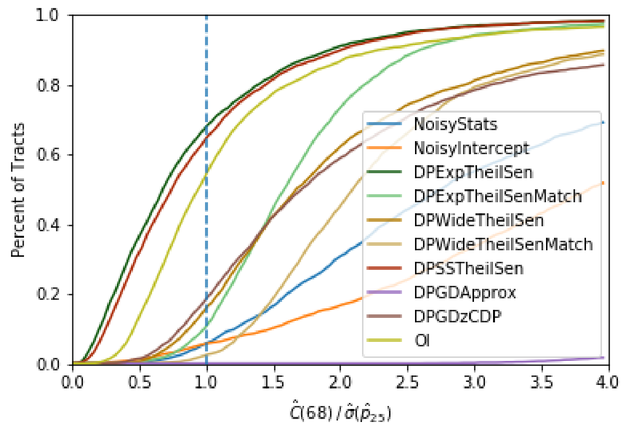
Additionally, while the methods used by the Opportunity Insights team are highly tailored to their setting relying on coordination across Census tracts, in order to compute an upper bound on the local sensitivity, as discussed in Section 1.3, the formally differentially private methods are general-purpose and do not require coordination across tracts.

Figures 7 shows empirical cumulative density distributions of the empirical error bounds on the 288 datasets in the Bikeshare dataset. Figure 8 shows the empirical cumulative density function of the output distribution on the Stockexchange dataset. Note that this is a different form to the empirical CDFs which appear in Figure 6 and Figure 7. Figure 9a shows the empirical cumulative density function of the output distribution on the Carbon Nanotubes dataset. Figures 7, 8 and 9a show that on the three other real datasets, for a range of realistic ε values the additional noise due to privacy, in particular using `DPExpTheilSen`, is less than the standard error.

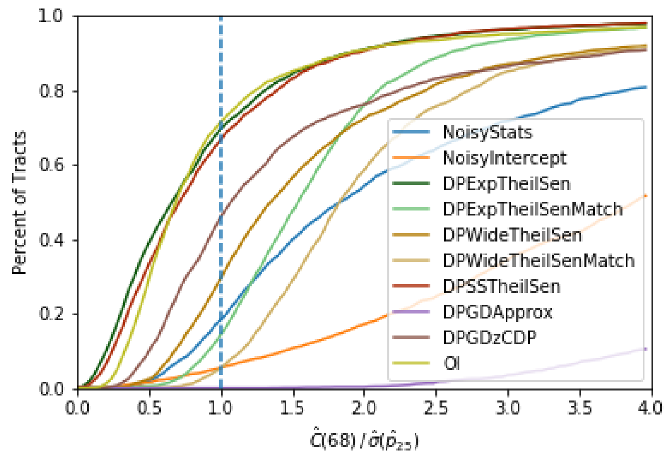
4.2 Robustness vs. Non-robustness: Guidance for Algorithm Selection

The DP algorithms we evaluate can be divided into two classes, *robust* DP estimators based on Theil-Sen — `DPSSTheilSen`, `DPExpTheilSen` and `DPWideTheilSen` — and *non-robust* DP estimators based on OLS — `NoisyStats` and `GradDescent`. Experimentally, we found that the algorithm's behaviour tend to be clustered in these two classes, with the robust estimators outperforming the non-robust estimators in a wide variety of parameter regimes. In most of the experiments we saw in the previous section (Figures 6, 7, 8 and, 9a), `DPExpTheilSen` was the best performing algorithm, followed by `DPWideTheilSen` and `DPSSTheilSen`. However, below we will see that in experiments on synthetic data, we found that the non-robust estimators outperform the robust estimators in some parameter regimes.

4.2.1 `NoisyStats` *and* `DPExpTheilSen`*.* In Figure 10, we investigate the relative performance ($\hat{C}_{\text{true}}(68)/\sigma(\widehat{p}_{25})$) of the algorithms in several parameter regimes of n, ε, and σ_x^2 on synthetically generated data. For each parameter setting, for each algorithm, we plot of the average value of $\hat{C}_{\text{true}}(68)/\sigma(\widehat{p}_{25})$ over 50 trials on a single dataset, and average again over 500 independently sampled



(a) Illinois (IL)



(b) North Carolina (NC)

Figure 6: Empirical CDF for the Empirical 68% error bounds, $\hat{C}(68)$, normalized by empirical OLS standard error when evaluated on Opportunity Insights data for the states of Illinois and North Carolina. The algorithm denoted by OI refers to the heuristic algorithm used in the deployment of the Opportunity Altas Tool [10]. Privacy parameter $\varepsilon = 16$ for the pair (p_{25}, p_{75}).

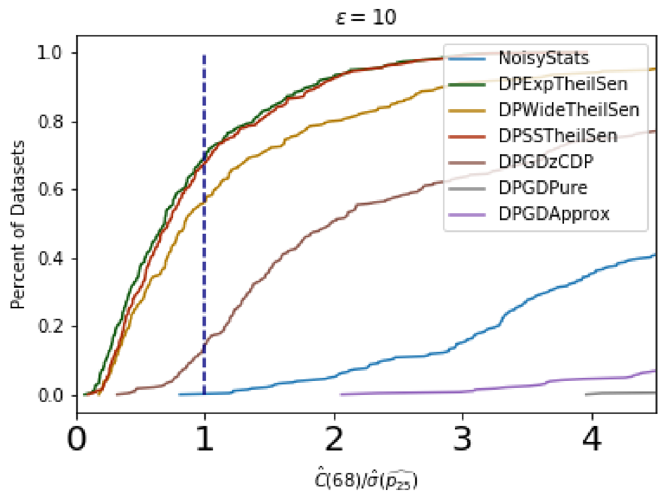


Figure 7: Bikeshare data. Empirical CDFs of the performance over the set of datasets when $\varepsilon = 10$.

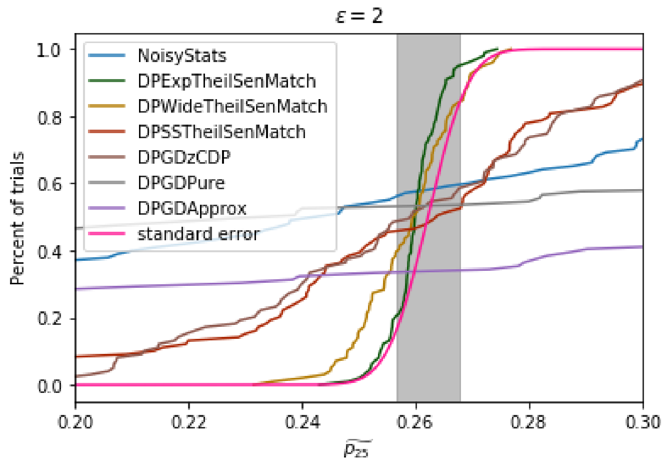
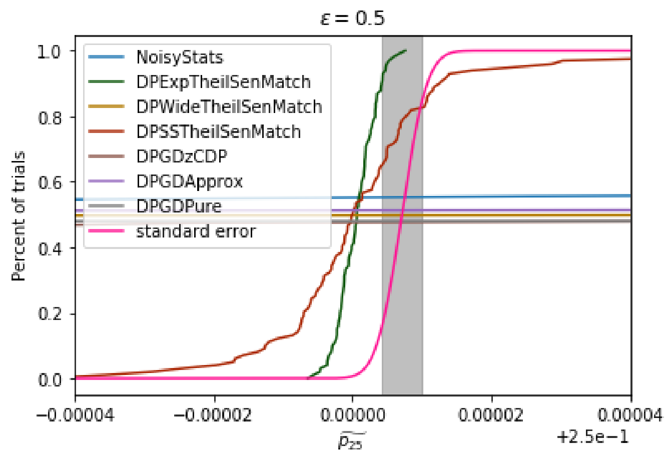


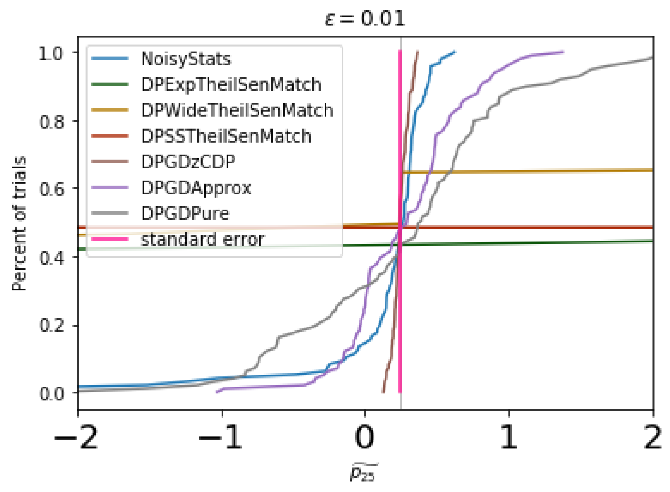
Figure 8: Stock Exchange UCI Data. Empirical cdf of the output distribution of the estimate of p_{25} after 100 trials of each algorithm with $\varepsilon = 2$. The grey region includes all the values that are within one standard error of $\widehat{\sigma}(\widehat{p}_{25})$. The curve labelled “standard error“ shows the non-private posterior belief on the value of the p_{25} assuming Gaussian noise.

datasets. Across large ranges for each of these three parameters ($n \in [30, 10,000]$; $\sigma_x^2 \in [0.003, 0.03]$; $\varepsilon \in [0.01, 10]$, all varied on a logarithmic scale), we see that either `DPExpTheilSen`, `DPGDzCDP`, or `NoisyStats` is consistently the best performing algorithm—or close to it. Note that of these three algorithms, only `DPExpTheilSen` and `NoisyStats` fulfill the stronger pure $(\varepsilon, 0)$-DP privacy guarantee. We see that `DPGDzCDP` and `NoisyStats` trend towards taking over as the best algorithms as $\varepsilon n \sigma_x^2$ increases.

For parameter regimes in which the non-robust algorithms outperform the robust estimators, `NoisyStats` is preferable to `DPGDzCDP` since it is more efficient, requires no hyperparameter tuning (except bounds on the inputs x_i and y_i), fulfills a stronger privacy guarantee, and releases the noisy sufficient statistics with no additional cost in privacy. Experimentally, we find that the main indicator for deciding between robust estimators and non-robust estimators is the quantity $\varepsilon n \text{var}(\mathbf{x})$ (which is a proxy for $\varepsilon n \sigma_x^2$). Roughly, when $\varepsilon n \text{var}(\mathbf{x})$ and σ_e^2 are both large, `NoisyStats` is close to optimal among the DP algorithms tested; otherwise, the robust estimator `DPExpTheilSen` typically has lower error. Hyperparameter tuning and the quantity $|x_{new} - \bar{x}|$ also play minor roles in determining the relative performance of the algorithms.



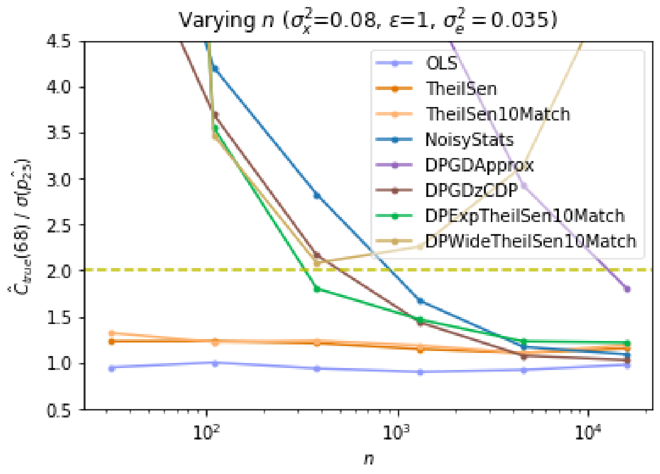
(a) Domain knowledge that $p_{25}, p_{75} \in [-0.5, 1.5]$



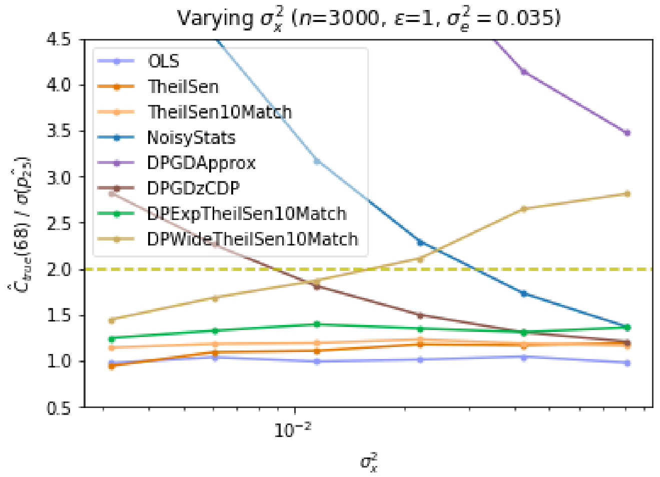
(b) Domain knowledge that $p_{25}, p_{75} \in [-50, 50]$

Figure 9: Carbon nanotubes UCI Data. Empirical cdf of the output distribution of the estimate of p_{25} after 100 trials of each algorithm. The grey region includes all the values that are within one standard error of $\widehat{p}_{25}$. The curve labelled "standard error" shows the non-private posterior belief on the value of the p_{25} assuming Gaussian noise.

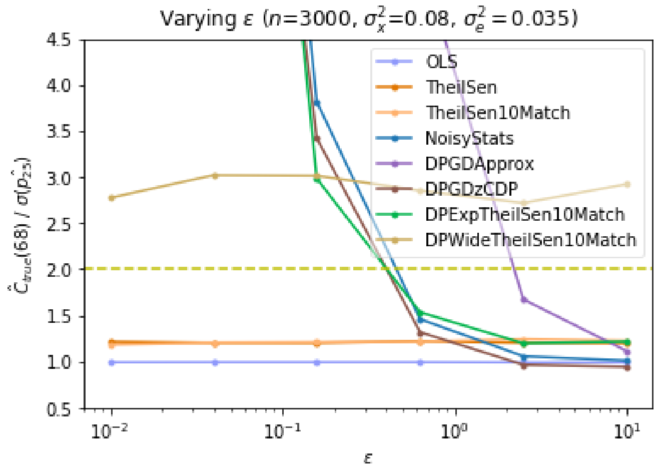
4.2.2 Role of εvar$(\mathbf{x})$. The quantity εvar$(\mathbf{x})$ is related to the size of the dataset, how concentrated the independent variable of the data is and how private the mechanism is. It appears to be a better indicator of the performance of DP mechanisms than any of the individual statistics ε, n, or var$(\mathbf{x})$ in isolation. In Figure 11 we compare the performance of `NoisyStats` and `DPExpTheilSen10Match` as we vary σ_e^2 and x_{new}. For each parameter setting, for each algorithm, the error is computed as the average error over 20 trials and 500 independently sampled datasets. Notice that the blue line presents the error of the non-private OLS estimator, which is our baseline. The quantity we control in these experiments is $\varepsilon n \sigma_x^2$, although we expect this to align closely with εvar$(\mathbf{x})$, which is the empirically measurable quantity. In all of our synthetic data



(a) Varying n

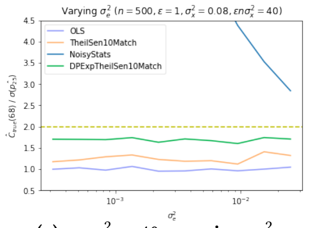


(b) Varying σ_x^2

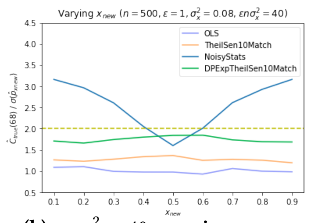


(c) Varying ε

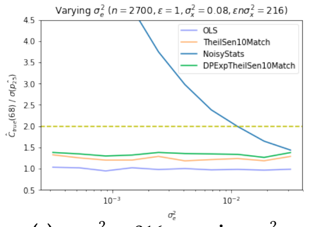
Figure 10: Relative error ($\hat{C}_{\textbf{true}}/\sigma(\widehat{p}_{25})$) of DP and non-private algorithms on synthetic data..



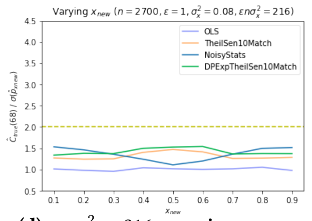
(a) $\varepsilon n\sigma_x^2 = 40$; varying σ_e^2



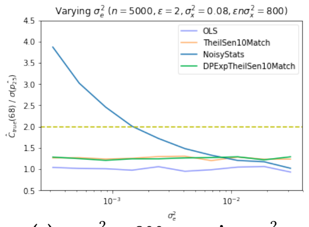
(b) $\varepsilon n\sigma_x^2 = 40$; varying x_{new}



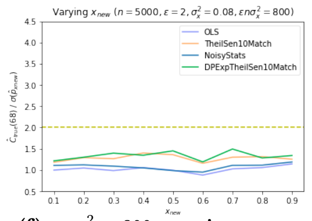
(c) $\varepsilon n\sigma_x^2 = 216$; varying σ_e^2



(d) $\varepsilon n\sigma_x^2 = 216$; varying x_{new}



(e) $\varepsilon n\sigma_x^2 = 800$; varying σ_e^2



(f) $\varepsilon n\sigma_x^2 = 800$; varying x_{new}

Figure 11: Comparing NoisyStats and DPExpTS10Match on synthetic data as σ_e^2 and x_{new} vary, for $\bar{x} = 0.5$. Plotting $\hat{C}_{\text{true}}/\sigma(\widehat{p}_{25})$. OLS and TheilSen10Match included for reference.

experiments, in which the x_i's are uniform and the e_i's are Gaussian, once $\varepsilon n\sigma_x^2 > 400$ and $\sigma_e^2 \geq 10^{-2}$, NoisyStats is close to the best performing algorithm. It is also important to note that once εnvar($\mathbf{x}$) is large, both NoisyStats and DPExpTheilSen perform well. See Figure 11 for a demonstration of this on the synthetic data.

The error, as measured by $\hat{C}_{\text{true}}(68)$, of both OLS and Theil-Sen estimators converges to 0 as $n \to \infty$ at the same asymptotic rate. However, OLS converges a constant factor faster than Theil-Sen resulting in its superior performance on relatively large datasets. As εnvar($\mathbf{x}$) increases, the error of NoisyStats decreases, and the outputs of this algorithm concentrate around the OLS estimate. While the outputs of DPExpTheilSen tend to be more concentrated, they converge to the Theil-Sen estimate, which has higher sampling error. Thus, as we increase εnvar($\mathbf{x}$), eventually the additional noise due to privacy added by NoisyStats is less than the difference between the OLS and Theil-Sen point estimates, resulting in NoisyStats outperforming DPExpTheilSen. This phenomenon can be seen in Figure 10 and Figure 11.

The classifying power of εnvar($\mathbf{x}$) is a result of its impact on the performance of NoisyStats. Recall that NoisyStats works by using noisy sufficient statistics within the closed form solution for OLS given in Equation 2. The noisy sufficient statistic nvar($\mathbf{x}$) + Lap$(0, 3\Delta_2/\varepsilon)$, appears in the denominator of this closed form solution. When εnvar($\mathbf{x}$) is small, this noisy sufficient statistic has constant mass concentrated near, or below, 0, and hence the inverse, $1/(\text{nvar}(\mathbf{x}) + \text{Lap}(0, 3\Delta_2/\varepsilon))$, which appears in the NoisyStats, can be an arbitrarily bad estimate of $1/\text{nvar}(\mathbf{x})$. In contrast, when εnvar($\mathbf{x}$) is large, the distribution of nvar($\mathbf{x}$) + Lap$(0, 3\Delta_2/\varepsilon)$ is more concentrated around nvar($\mathbf{x}$), so that with high probability $1/(\text{nvar}(\mathbf{x}) + \text{Lap}(0, 3\Delta_2/\varepsilon))$ is close to $1/\text{nvar}(\mathbf{x})$.

In Figures 10a, 10b and 10c, the performance of all the algorithms, including NoisyStats and DPExpTheilSen, are shown as we vary each of the parameters ε, n and σ_x^2, while holding the other variables constant, on synthetic data. In doing so, each of these plots is effectively varying εnvar($\mathbf{x}$) as a whole. These plots suggest that all three parameters help determine which algorithm is the most accurate. Figure 11 confirms that εnvar($\mathbf{x}$) is a strong indicator of the relative performance of NoisyStats and DPExpTheilSen, even as other variables in the OLS standard error equation (3) – including the variance of the noise of the dependent variable, σ_e, and the difference between x_{new} and the mean of the x values, $|x_{new} - \bar{x}|$ – are varied.

4.2.3 *The Role of σ_e^2.*

One main advantage that all the DPTheilSen algorithms have over NoisyStats is that they are better able to adapt to the specific dataset. When σ_e^2 is small, there is a strong linear signal in the data that the non-private OLS estimator and the DPTheilSen algorithms can leverage. Since the noise addition mechanism of NoisyStats does not leverage this strong signal, its relative performance is worse when σ_e^2 is small. Thus, σ_e^2 affects the relative performance of the algorithms, even when εnvar($\mathbf{x}$) is large. We saw this effect in Figure 9a on the Carbon Nanotubes UCI data, where εnvar$(x) = 889.222$ is large, but NoisyStats performed poorly relative to the DPTheilSen algorithms.

In each of the plots 11a, 11c, and 11e, the quantity εnvar($\mathbf{x}$) is held constant while σ_e^2 is varied. These plots confirm that the performance of NoisyStats degrades, relative to other algorithms, when σ_e^2 is small. As σ_e^2 increases, the relative performance of NoisyStats improves until it drops below the relative performance of the TheilSen estimates. The cross-over point varies with εnvar($\mathbf{x}$). In Figure 11a, we see that when εnvar($\mathbf{x}$) is small, the methods based on Theil-Sen are the better choice for all values of σ_e^2 studied. When we increase εnvar($\mathbf{x}$) in Figure 11e we see a clear cross over point. These plots add evidence to our conjecture that robust estimators are a good choice when εnvar($\mathbf{x}$) and σ_e^2 are both small, while non-robust estimators perform well when εnvar$\mathbf{x}$ is large.

4.2.4 *The Role of $|x_{new} - \bar{x}|$.*

The final factor we found that plays a role in the relative performance of NoisyStats and DPExpTheilSen is $|x_{new} - \bar{x}|$. This effect is less pronounced than that of εnvar($\mathbf{x}$) or σ_e^2, and seems to rarely change the ordering of DP algorithms. In Figures 11b, 11d and 11f, we explore the effect of this quantity on the relative performance of OLS, TheilSen10Match, DPExpTheilSen10Match, and NoisyStats. We saw in Equation 3 that $|x_{new} - \bar{x}|$ affects the standard error $\hat{\sigma}(\widehat{p}_{25})$ - the further x_{new} is from the centre of the data, $\bar{x}$, the less certain we are about the OLS point estimate. All algorithms have better performance when $|x_{new} - \bar{x}|$ is small; however, this effect is most pronounced with NoisyStats. In NoisyStats, $\widetilde{p}_{x_{new}} = \tilde{\alpha}(x_{new} - \bar{x}) + \bar{y} + L_3$ where $L_3 \sim \text{Lap}(0, 3(1 + |\tilde{\alpha}|)/(\varepsilon n))$ and $\tilde{\alpha}$ and $\tilde{\beta}$ are the noisy slope and intercepts respectively. Thus, we expect a large $|x_{new} - \bar{x}|$ to amplify the error present in $\tilde{\alpha}$. We vary x_{new} between 0 and 1. As expected, the error is minimized when $x_{new} = 0.5$, since $\bar{x} = 0.5$. Since we expect the error in $\tilde{\alpha}$ to decrease as we increase $\varepsilon n\text{var}(\mathbf{x})$, this explains why the quantity $|x_{new} - \bar{x}|$ has a larger effect when $\varepsilon n\text{var}(\mathbf{x})$ is small.

4.2.5 The Role of Hyperparameter Tuning. A final major distinguishing feature between the DPTheilSen algorithms, NoisyStats and DPGradDescent is the amount of prior knowledge needed by the data analyst to choose the hyperparameters appropriately. Notably, NoisyStats does not require any hyperparameter tuning other than a bound on the data. The DPTheilSen algorithms require some reasonable knowledge of the range that p_{25} and p_{75} lie in in order to set r_l and r_u. Finally, DPGradDescent requires some knowledge of where the input values lie, so it can set τ and T.

In Figure 9b, NoisyStats and all three DPGradDescent algorithms outperform the robust estimators. This experiment differs from Figure 9a in two important ways: the privacy parameter ε has decreased from 1 to 0.01, and the feasible output region for the DP TheilSen methods has increased from $[-0.5, 1.5]$ to $[-50, 50]$. When ε is large, the DPTheilSen algorithms are robust to the choice of this region since any area outside the range of the data is exponentially down weighted (see Figure 1). However, when ε is small, the size of this region can have a large effect on the stability of the output. As ε decreases, the output distributions of the DPTheilSen estimators are flattened, so that they are essentially sampling from the uniform distribution on the range of the parameter. This effect likely explains the poor performance of the robust estimators in Figure 9b, and highlights the importance of choosing hyperparameters carefully. If ε is small (ε much less than 1) and the analyst does not have a decent estimate of the range of p_{25}, then NoisyStats may be a safer choice than DPTheilSen.

4.3 Which robust estimator?

In the majority of the regimes we have tested, DPExpTheilSen outperforms all the other private algorithms. While DPSSTheilSen can be competitive with DPExpTheilSen and DPWideTheilSen, it rarely seems to outperform them. However, DPWideTheilSen can significantly outperform DPExpTheilSen when the standard error is small. Figure 12 compares the performance of DPExpTheilSen and DPWideTheilSen on the Bikeshare UCI data. When there is little noise in the data we expect the set of pairwise estimates that TheilSen takes the median of to be highly concentrated. We discussed in Section 2.2 why this is a difficult setting for DPExpTheilSen: in the continuous setting, the exponential mechanism based median algorithm can fail to put sufficient probability mass around the median, even if the data is concentrated at the median (see Figure 1). DPWideTheilSen was designed exactly as a fix to this problem. The parameter θ needs to be chosen carefully. In Figure 9a, θ is set to be much larger than the standard error, resulting in DPWideTheilSen performing poorly.

4.4 Which non-robust estimator?

There are two main non-robust estimators we consider: DPGradDescent and NoisyStats. DPGradDescent has three versions – DPGDPure, DPGDApprox, and DPGDzCDP – corresponding to the pure, approximate, and zero-concentrated variants. Amongst the DPGradDescent

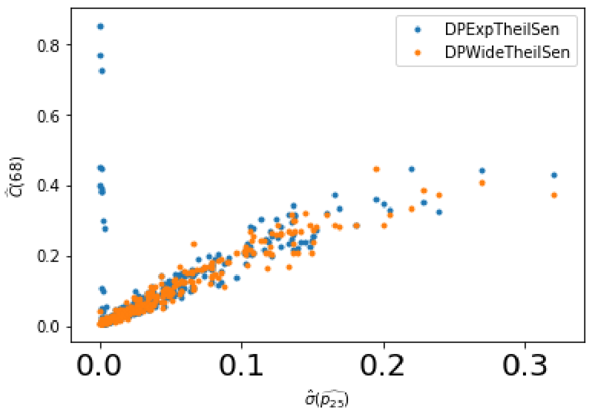


Figure 12: Comparison of DPExpTheilSen and DPWideExpTheilSen on Bikeshare UCI data with $\varepsilon = 2$ and $\theta = 0.01$. Datasets are sorted by their standard errors. For each dataset, there is a dot corresponding to the $\hat{C}(68)$ value of each algorithm. Both dots lie on the same vertical line.

algorithms, as expected, DPGDzCDP provides the best utility followed by DPGDApprox, and then DPGDPure. But how do these compare to NoisyStats? NoisyStats outperforms both DPGDPure and DPGDApprox for small δ (e.g. $\delta = 2^{-30}$ in our experiments). DPGDzCDP consistently outperforms NoisyStats, but the gap in performance is small in the regime where the non-robust estimators beat the robust estimators. Moreover, NoisyStats achieves a stronger privacy guarantee (pure $(\varepsilon, 0)$-DP rather than $\varepsilon^2/2$-zCDP). A fairer comparison is to use the natural $\varepsilon^2/2$-zCDP analogue of NoisyStats (using Gaussian noise and zCDP composition), in which case we have found that the advantage of DPGDzCDP significantly shrinks and in some cases is reversed. (Experiments omitted.)

The performance of the DPGradDescent algorithms also depend on hyperparameters that need to be carefully tuned, such as the number of gradient calls T and the clip range $[-\tau, \tau]$. Since NoisyStats requires less hyperparameters, this makes DPGradDescent potentially harder to use in practice. In addition, NoisyStats is more efficient and can be used to release the noisy sufficient statistics with no additional privacy loss. Since the performance of the two algorithms is similar in the regime where non-robust methods appear to have more utility than the robust ones, the additional benefits of NoisyStats may make it preferable in practice.

We leave a more thorough evaluation and optimization of these algorithms in the regime of large n, including how to optimize the hyperparameters in a DP manner, to future work.

4.5 Analyzing the bias

Let $(p_{25}^{TS}, p_{75}^{TS})$ be the prediction estimates produced using the non-private TheilSen estimator. The non-robust DP methods – NoisyStats and DPGradDescent – approach $(\widehat{p}_{25}, \widehat{p}_{75})$ as $\varepsilon \to \infty$, while the DPTheilSen methods approach $(p_{25}^{TS}, p_{75}^{TS})$ as $\varepsilon \to \infty$. For any fixed dataset, $(\widehat{p}_{25}, \widehat{p}_{75})$ and $(p_{25}^{TS}, p_{75}^{TS})$ are not necessarily equal. A good representation of this bias can be seen in Figure 9a.

However, both the TheilSen estimator and the OLS estimator are consistent unbiased estimators of the true slope in simple linear regression. That is, as $n \longrightarrow \infty$, both $(p_{25}^{TS}, p_{75}^{TS})$ and $(\widehat{p}_{25}, \widehat{p}_{75})$ tend to the true value (p_{25}, p_{75}). Thus, all the private algorithms output the true prediction estimates as $n \longrightarrow \infty$, for a fixed ε.

5 CONCLUSION

It is possible to design DP simple linear regression algorithms where the distortion added by the private algorithm is less than the standard error, even for small datasets. In this work, we found that in order to achieve this we needed to switch from OLS regression to the more robust linear regression estimator, Theil-Sen. We identified key factors that analysts should consider when deciding whether DP methods based on robust or non-robust estimators are right for their application.

ACKNOWLEDGMENTS

We thank Raj Chetty, John Friedman, and Daniel Reuter from Opportunity Insights for many motivating discussions, providing us with simulated Opportunity Atlas data for our experiments, and helping us implement the OI algorithm. We also thank Cynthia Dwork and James Honaker for helpful conversations at an early stage of this work; Garett Bernstein and Dan Sheldon for discussions about their work on private Bayesian inference for linear regression; Sofya Raskhodnikova, Thomas Steinke, and others at the Simons Institute Spring 2019 program on data privacy for helpful discussions on specific technical aspects of this work.

REFERENCES

[1] 2013. A novel Hybrid RBF Neural Networks model as a forecaster. *Statistics and Computing* (2013). https://doi.org/10.1007/s11222-013-9375-7

[2] Jacob Abernethy, Chansoo Lee, and Ambuj Tewari. 2016. Perturbation techniques in online learning and optimization. *Perturbations, Optimization, and Statistics* (2016), 233.

[3] Mehmet Aci and Mutlu Avci. 2016. Artificial Neural Network Approach for Atomic Coordinate Prediction of Carbon Nanotubes. *Applied Physics A* 122, 631 (2016).

[4] Raef Bassily, Adam D. Smith, and Abhradeep Thakurta. 2014. Private Empirical Risk Minimization, Revisited. *CoRR* abs/1405.7085 (2014). http://arxiv.org/abs/1405.7085

[5] Garrett Bernstein and Daniel R Sheldon. 2019. Differentially Private Bayesian Linear Regression. In *Advances in Neural Information Processing Systems 32.* 523–533.

[6] Mark Bun and Thomas Steinke. 2016. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference.* Springer, 635–658.

[7] Mark Bun and Thomas Steinke. 2019. Average-Case Averages: Private Algorithms for Smooth Sensitivity and Mean Estimation. *arXiv preprint arXiv:1906.02830* (2019).

[8] Tony Cai, Yichen Wang, and Linjun Zhang. 2019. The Cost of Privacy: Optimal Rates of Convergence for Parameter Estimation with Differential Privacy. *CoRR* abs/1902.04495 (2019). http://arxiv.org/abs/1902.04495

[9] Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. 2011. Differentially Private Empirical Risk Minimization. *Journal of Machine Learning Research* 12 (2011), 1069–1109.

[10] Raj Chetty and John N. Friedman. 2019. A Practical Method to Reduce Privacy Loss When Disclosing Statistics Based on Small Samples. *American Economic Review Papers and Proceedings* 109 (2019), 414–420.

[11] Raj Chetty, John N Friedman, Nathaniel Hendren, Maggie R Jones, and Sonya R Porter. 2018. *The opportunity atlas: Mapping the childhood roots of social mobility.* Technical Report. National Bureau of Economic Research.

[12] Raj Chetty, Nathaniel Hendren, Patrick Kline, and Emmanuel Saez. 2014. Where is the land of opportunity? The geography of intergenerational mobility in the United States. *The Quarterly Journal of Economics* 129, 4 (2014), 1553–1623.

[13] Graham Cormode. [n.d.]. Building Blocks of Privacy: Differentially Private Mechanisms. ([n. d.]), 18–19.

[14] Simon Couch, Zeki Kazan, Kaiyan Shi, Andrew Bray, and Adam Groce. 2019. Differentially Private Nonparametric Hypothesis Testing. *arXiv preprint arXiv:1903.09364* (2019).

[15] Irit Dinur and Kobbi Nissim. 2003. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems.* 202–210.

[16] Cynthia Dwork and Jing Lei. 2009. Differential privacy and robust statistics.. In *STOC*, Vol. 9. 371–380.

[17] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings.* 265–284.

[18] Cynthia Dwork, Kunal Talwar, Abhradeep Thakurta, and Li Zhang. 2014. Analyze gauss: optimal bounds for privacy-preserving principal component analysis. In *STOC.* 11–20.

[19] Hadi Fanaee-T and Joao Gama. 2013. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence* (2013), 1–15. https://doi.org/10.1007/s13748-013-0040-3

[20] Frank McSherry and Kunal Talwar. 2007. Mechanism Design via Differential Privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings.* 94–103.

[21] Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. 2007. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007.* 75–84.

[22] Pranab Kumar Sen. 1968. Estimates of the regression coefficient based on Kendall's tau. *Journal of the American statistical association* 63, 324 (1968), 1379–1389.

[23] Or Sheffet. 2017. Differentially Private Ordinary Least Squares. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017.* 3105–3114. http://proceedings.mlr.press/v70/sheffet17a.html

[24] Henri Theil. 1950. A rank-invariant method of linear and polynomial regression analysis, 3; confidence regions for the parameters of polynomial regression equations. *Indagationes Mathematicae* 1, 2 (1950), 467–482.

[25] Yu-Xiang Wang. 2018. Revisiting differentially private linear regression: optimal and adaptive prediction & estimation in unbounded domain. *CoRR* abs/1803.02596 (2018). arXiv:1803.02596 http://arxiv.org/abs/1803.02596

[26] Xin Yan and Xiao Gang Su. 2009. *Linear Regression Analysis: Theory and Computing.* World Scientific Publishing Co., Inc., USA.

A OPPORTUNITY INSIGHTS APPLICATION

A.1 Data Generating Process for OI Synthetic Datasets

In this subsection, we describe the data generating process for simulating census microdata from the Opportunity Insights team. In the model, assume that each set of parents i in tract t and state m has one child (also indexed by i).

Size of Tract: The size of each tract in any state is a random variable. Let $\text{Exp}(b)$ represent the exponential distribution with scale b. Then if n_{tm} is the number of people in tract t and state m, then $n_{tm} \sim \lfloor \text{Exp}(52) + 20 \rfloor$. This distribution over simulated counts was chosen by the Opportunity Insights team because it closely matches the distribution of tract sizes in the real data.

Linear Income Relationship: Let x_{itm} be the child income given the parent income y_{itm}, then we enforce the following relationship between x_{itm} and y_{itm}:

$$\ln(x_{itm}) = \alpha_{tm} + \beta_{tm} \ln(y_{itm}) + e_i,$$

where $e_i \sim \mathcal{N}(0, (0.20)^2)$ and α_{tm}, β_{tm} are calculated from public estimates of child income rank given the parent income rank. [3] Next, p_{im} is calculated as the parent i's percentile income within

[3] Gleaned from some tables the Opportunity Insights team publicly released with tract-level (micro) data. See https://opportunityatlas.org/. The dataset used to calculate the α_{tm}, β_{tm} is aggregated from some publicly-available income data for all tracts within all U.S. states between 1978 and 1983.

the state m's parent income distribution (and rounded up to the 2nd decimal place).

Parent Income Distribution: Let μ_{tm} denote the public estimate of the mean household income for tract t in state m (also obtained from publicly-available income data used to calculate α_{tm}, β_{tm}). Empirically, the Opportunity Insights team found that the within-tract standard deviation of parent incomes is about twice the between-tract standard deviation. Let $\mathrm{Var}(\mu_{tm})$ denote the sample variance of the estimator μ_{tm}. Then enforce that $\mathrm{Var}_{tm}(y_{itm}) = 4\,\mathrm{Var}(\mu_{tm})$ where $\mathrm{Var}_{tm}(y_{itm})$ is the variance of parental income y_{itm} in tract t and state m. Furthermore, assume that y_{itm} are log-normally distributed within each tract and draw $\ln(y_{itm})$ from $\mathcal{N}(\mathbb{E}_{tm}[\ln(y_{itm})], \mathrm{Var}_{tm}[\ln(y_{itm})])$ for $i = 1, \ldots, n_{tm}$, where

$$\mathbb{E}_{tm}[\ln(y_{itm})] = 2\ln(\mu_{tm}) - 0.5\ln(\underset{tm}{\mathrm{Var}}(y_{itm}) + \mu_{tm}^2),$$

and

$$\underset{tm}{\mathrm{Var}}[\ln(y_{itm})] = -2\ln(\mu_{tm}) + \ln(\underset{tm}{\mathrm{Var}}(y_{itm}) + \mu_{tm}^2).$$

A.2 The Maximum Observed Sensitivity Algorithm

Opportunity Insights [10] provided a practical method – which they term the "Maximum Observed Sensitivity" (MOS) algorithm – to reduce the privacy loss of their released estimates. This method is not formally differentially private. We use MOS or OI interchangeably to refer to their statistical disclosure limitation method. The crux of their algorithm is as follows: The maximum observed sensitivity, corresponding to an upper envelope on the largest local sensitivity across tracts in a state in the dataset, is calculated and Laplace noise of this magnitude divided by the number of people in that cell is added to the estimate and then released. The statistics they release include 0.25, 0.75 percentiles per cell, the standard error of these percentiles, and the count.

Two notes are in order. First, the MOS algorithm does not calculate the local sensitivity exactly but uses a lower bound by overlaying an 11×11 grid on the $[0, 1] \times [0, 1]$ space of possible $\mathbf{x}, \mathbf{y}$ values. Then the algorithm proceeds to add a datapoint from this grid to the dataset and calculate the maximum change in the statistic, which is then used as a lower bound. Second, analysis are performed only on census tracts that satisfy the following property: they must have at least 20 individuals with 10% of parent income percentiles in that tract above the state parent income median percentile and 10% below. If a tract does not satisfy this condition then no regression estimate is released for that tract.

B SOME RESULTS IN DIFFERENTIAL PRIVACY

In this section we will briefly review some of the fundamental definitions and results pertaining to general differentially private algorithms.

For any query function $f : \mathcal{X}^n \to \mathbb{R}^K$ let

$$\mathrm{GS}_f = \max_{d \sim d'} \|f(d) - f(d')\|,$$

called the global sensitivity, be the maximum amount the query can differ on neighboring datasets.

Theorem 12 (Laplace Mechanism [17]). *For any privacy parameter $\varepsilon > 0$ and any given query function $f : \mathcal{X}^n \to \mathbb{R}^K$ and database $d \in \mathcal{X}^n$, the* Laplace mechanism *outputs*

$$\tilde{f}_L(d) = f(d) + (R_1, \ldots, R_K),$$

where $R_1, \ldots, R_K \sim Lap(0, \frac{GS_f}{\varepsilon})$ are i.i.d. random variables drawn from the 0-mean Laplace distribution with scale $\frac{GS_f}{\varepsilon}$.

The Laplace mechanism is $(\varepsilon, 0)$-DP.

Theorem 13 (Exponential Mechanism [20]). *Given an arbitrary range $\mathcal{R}$, let $u : \mathcal{X}^n \times \mathcal{R} \to \mathbb{R}$ be a utility function that maps database/output pairs to utility scores. Let $GS_u = \max_r GS_{u(\cdot, r)}$. For a fixed database $d \in \mathcal{X}^n$ and privacy parameter $\varepsilon > 0$, the* exponential mechanism *outputs an element $r \in \mathcal{R}$ with probability proportional to $\exp\left(\frac{\varepsilon \cdot u(d,r)}{2GS_u}\right)$.*

The exponential mechanism is $(\varepsilon, 0)$-DP.

The following results allow us to use differentially private algorithms as building blocks in larger algorithms.

Lemma 14 (Post-Processing [17]). *Let $M : \mathcal{X}^n \to \mathcal{Y}$ be an (ε, δ) differentially private and $f : \mathcal{Y} \to \mathcal{R}$ be a (randomized) function. Then $f \circ M : \mathcal{X}^n \to \mathcal{R}$ is an (ε, δ) differentially private algorithm.*

Theorem 15 (Basic Composition [17]). *For any $k \in [K]$, let M_k be an $(\varepsilon_k, \delta_k)$ differentially private algorithm. Then the composition of the T mechanisms $M = (M_1, \ldots, M_K)$ is (ε, δ) differentially private where $\varepsilon = \sum_{k \in [K]} \varepsilon_k$ and $\delta = \sum_{k \in [K]} \delta_k$.*

Definition 16 (Coupling). *Let $\boldsymbol{z}$ and $\boldsymbol{z}'$ be two random variables defined over the probability spaces Z and Z', respectively. A coupling of $\boldsymbol{z}$ and $\boldsymbol{z}'$ is a joint variable $(\boldsymbol{z}_c, \boldsymbol{z}'_c)$ taking values in the product space $(Z \times Z')$ such that $\boldsymbol{z}_c$ has the same marginal distribution as $\boldsymbol{z}$ and $\boldsymbol{z}'_c$ has the same marginal distribution as $\boldsymbol{z}'$.*

Definition 17 (c-Lipschitz randomized transformations). *A randomized transformation $T : \mathcal{X}^n \to \mathcal{Y}^m$ is c-Lipschitz if for all datasets $d, d' \in \mathcal{X}^n$, there exists a coupling $(\boldsymbol{z}_c, \boldsymbol{z}'_c)$ of the random variables $\boldsymbol{z} = T(d)$ and $\boldsymbol{z}' = T(d')$ such that with probability 1,*

$$H(\boldsymbol{z}_c, \boldsymbol{z}'_c) \leq c \cdot H(d, d')$$

where H denotes Hamming distance.

Lemma 18 (Composition with Lipschitz transformations (well–known)). *Let M be an (ε, δ)-DP algorithm, and let T be a c-Lipschitz transformation of the data with respect to the Hamming distance. Then, $M \circ T$ is $(c\varepsilon, \delta)$-DP.*

Proof. Let $H(d, d')$ denote the distance (in terms of additions and removals or swaps) between datasets d and d'. By definition of the Lipschitz property, $H(T(d), T(d')) \leq c \cdot H(d, d')$. The lemma follows directly from the Lipschitz property on adjacent databases and the definition of (ε, δ)-differential privacy. □

C NOISYSTATS

C.1 Privacy Proof of NoisyStats

Lemma 19. *We are given two vectors $\boldsymbol{x}, \boldsymbol{y} \in [0, 1]^n$. Let*

$$ncov(\boldsymbol{x}, \boldsymbol{y}) = \langle \boldsymbol{x} - \bar{x}\mathbf{1}, \boldsymbol{y} - \bar{y}\mathbf{1} \rangle = (\sum_{i=1}^{n} x_i \cdot y_i) - \frac{(\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n}$$

and

$$nvar(\boldsymbol{x}) = \langle \boldsymbol{x}-\bar{x}\mathbf{1}, \boldsymbol{x}-\bar{x}\mathbf{1}\rangle = (\sum_{i=1}^{n} x_i^2) - \frac{(\sum_{i=1}^{n} x_i)^2}{n}.$$

Also, let $\bar{x}, \bar{y}$ be the means of $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively and $\mathbf{1}$ be the all ones vector.

Then if GS_{ncov} and GS_{nvar} are the global sensitivities of functions ncov and nvar then $GS_{ncov} = \left(1 - \frac{1}{n}\right)$ and $GS_{nvar} = \left(1 - \frac{1}{n}\right)$.

Proof. Let $\mathbf{z} = \langle \mathbf{x}, \mathbf{y}\rangle$ and $\mathbf{z}' = \langle \mathbf{x}', \mathbf{y}'\rangle$ be neighbouring databases differing on the nth datapoint [4]. Let $a = \sum_{i=1}^{n-1} x_i$ and $b = \sum_{i=1}^{n-1} y_i$ and note that $\max\{a, b\} \le n-1$. Then,

$$\begin{aligned}\text{nvar}(\mathbf{x}) - \text{nvar}(\mathbf{x}') &= x_n^2 - x_n'^2 - \frac{2ax_n}{n} - \frac{x_n^2}{n} + \frac{2ax_n'}{n} + \frac{x_n'^2}{n} \\ &= (1-\frac{1}{n})(x_n^2 - x_n'^2) + \frac{2a}{n}(x_n' - x_n).\end{aligned}$$

If $x_n' - x_n \le 0$ then $\text{nvar}(\mathbf{x}) - \text{nvar}(\mathbf{x}') \le (1-\frac{1}{n})(x_n^2 - x_n'^2) \le 1 - \frac{1}{n}$. Otherwise,

$$\begin{aligned}\text{nvar}(\mathbf{x}) - \text{nvar}(\mathbf{x}') &\le (1-\frac{1}{n})(x_n^2 - x_n'^2) + \frac{2(n-1)}{n}(x_n' - x_n) \\ &= (1-\frac{1}{n})(x_n^2 - 2x_n + 2x_n' - x_n'^2)\end{aligned}$$

Since $x_n \in [0,1]$ we have $x_n^2 - 2x_n \in [-1, 0]$, so $\text{nvar}(\mathbf{x}) - \text{nvar}(\mathbf{x}') \le 1 - \frac{1}{n}$.

Also,

$$\begin{aligned}&\text{ncov}(\mathbf{x},\mathbf{y}) - \text{ncov}(\mathbf{x}',\mathbf{y}') \\ &= x_n y_n - x_n' y_n' + \\ &\frac{a(y_n' - y_n) + b(x_n' - x_n) + x_n' y_n' - x_n y_n}{n} \\ &\le (1-\frac{1}{n})(x_n y_n - x_n' y_n') + \\ &\frac{a(y_n' - y_n) + b(x_n' - x_n)}{n}\end{aligned}$$

If $y_n' - y_n \le 0$ and $x_n' - x_n \le 0$ then $\text{ncov}(\mathbf{x},\mathbf{y}) - \text{ncov}(\mathbf{x}',\mathbf{y}') \le (1-\frac{1}{n})(x_n y_n - x_n' y_n') \le (1-\frac{1}{n})$. If $y_n' - y_n \le 0$ and $x_n' - x_n > 0$ then $\text{ncov}(\mathbf{x},\mathbf{y}) - \text{ncov}(\mathbf{x}',\mathbf{y}') \le (1-\frac{1}{n})(x_n y_n - x_n' y_n' + (x_n' - x_n))$. Since $x_n y_n - x_n \le 0$ and $x_n' - x_n' y_n' \le 1$ we have $\text{ncov}(\mathbf{x},\mathbf{y}) - \text{ncov}(\mathbf{x}',\mathbf{y}') \le 1 - \frac{1}{n}$. Similarly if $y_n' - y_n > 0$ and $x_n' - x_n \le 0$ then $\text{ncov}(\mathbf{x},\mathbf{y}) - \text{ncov}(\mathbf{x}',\mathbf{y}') \le 1 - \frac{1}{n}$. Finally, if $y_n' - y_n > 0$ and $x_n' - x_n > 0$, we have

$$\begin{aligned}&\text{ncov}(\mathbf{x},\mathbf{y}) - \text{ncov}(\mathbf{x}',\mathbf{y}') \le \\ &(1-\frac{1}{n})(x_n y_n - x_n' y_n' + (y_n' - y_n) + (x_n' - x_n)) \\ &\le (1-\frac{1}{n})(x_n(y_n - 1) - x_n'(y_n' - 1) + (y_n' - 1) - (y_n - 1)) \\ &\le (1-\frac{1}{n})((x_n - 1)(y_n - 1) - (x_n' - 1)(y_n' - 1)).\end{aligned}$$

Since, $(x_n - 1)(y_n - 1) \in [0,1]$ and $(x_n' - 1)(y_n' - 1) \in [0,1]$, we have $\text{ncov}(\mathbf{x},\mathbf{y}) - \text{ncov}(\mathbf{x}',\mathbf{y}') \le 1 - \frac{1}{n}$. ☐

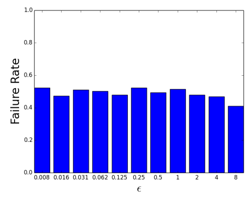
[4]This is without loss of generality as we can always "rotate" both databases until the index on which they differ becomes the nth datapoint.

Proof of Lemma 4: (NoisyStats). The global sensitivity of both $\text{ncov}(\mathbf{x},\mathbf{y})$ and $\text{nvar}(\mathbf{x})$ is bounded by $\Delta = (1 - 1/n)$ (by Lemma 19).

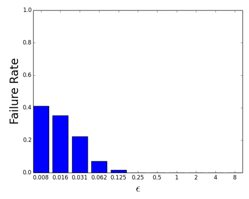
As a result, if we sample $L_1, L_2 \sim \text{Lap}(0, 3\Delta/\varepsilon)$ then both $\text{ncov}(\mathbf{x},\mathbf{y}) + L_1$ and $\text{nvar}(\mathbf{x}) + L_2$ are $(\varepsilon/3, 0)$-DP estimates by the Laplace mechanism guarantees (see Theorem 12). By the post-processing properties of differential privacy (Lemma 14), $1/(\text{nvar}(\mathbf{x}) + L_2)$ is a private release and the test $\text{nvar}(\mathbf{x}) + L_2 > 0$ is also private. As a result, $\tilde{\alpha}$ is a $(2\varepsilon/3, 0)$-DP release. Now to calculate the private intercept $\tilde{\beta}$, we use the global sensitivity of $(\bar{y} - \tilde{\alpha}\bar{x})$ which is at most $1/n \cdot (1 + |\tilde{\alpha}|)$, since the means of $\mathbf{x}, \mathbf{y}$ can change by at most $1/n$. [5] The Laplace noise we add ensures the private release of the intercept is $(\varepsilon/3, 0)$-DP.

Finally, by composition properties of differential privacy (Theorem 15), Algorithm 1 is $(\varepsilon, 0)$-DP. ☐

C.2 On the Failure Rate of `NoisyStats`

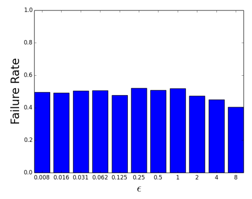


(a) $\hat{\alpha}$ = 0.45577, **nvar(x)** = 0.0245, n = 39
For Tract 800100 in County 31 in IL

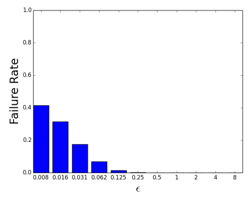


(b) $\hat{\alpha}$ = 0.25217, **nvar(x)** = 28.002, n = 381
For Tract 520100 in County 31 in IL

Figure 13



(a) $\hat{\alpha}$ = 0.0965, **nvar(x)** = 0.0245, n = 39
For Tract 2008 in County 63 in NC



(b) $\hat{\alpha}$ = 0.03757, **nvar(x)** = 31.154, n = 433
For Tract 603 in County 147 in NC

Figure 14

Unlike all the other algorithms in this paper, `NoisyStats` (Algorithm 1) can fail by returning $\perp$. It fails if and only if the noisy $\text{nvar}(\mathbf{x})$ sufficient statistic becomes 0 or negative i.e. $\text{nvar}(\mathbf{x}) + L_2 \le 0$ where $L_2 \sim \text{Lap}(0, 3(1 - 1/n)/\varepsilon)$. Since the statistic $\text{nvar}(\mathbf{x})$ will always be non-negative, we also require the private version of this statistic to be non-negative. Intuitively, we see that $\text{nvar}(\mathbf{x}) + L_2$ is more likely to be less than or equal to 0 if $\text{nvar}(\mathbf{x})$ is small or ε is small. The setting of ε directly affects the standard deviation

[5] Alternatively, to estimate $\tilde{\beta}$, one could compute $\tilde{x}$, $\tilde{y}$, private estimates of $\bar{x}$, $\bar{y}$ by adding Laplace noise from $\text{Lap}(0, 1/n)$ and then compute $\hat{\beta} = \tilde{y} - \hat{\alpha}\tilde{x}$.

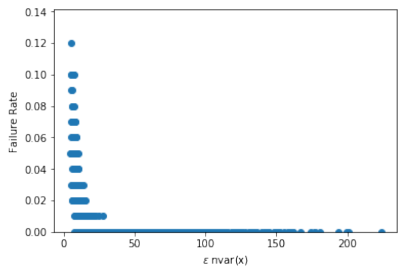


Figure 15: Failure rate of `NoisyStats` for all tracts in IL sorted by εnvar($\mathbf{x}$)

of the noise distribution added to ensure privacy. The smaller ε is, the more spread out the noise distribution is.So we would expect that when ε or nvar($\mathbf{x}$) is small, Algorithm 1 would fail more often. We experimentally show this observation holds on some tracts in IL and NC (from the semi-synthetic datasets given to us by the Opportunity Insights team).

In Figures 13a, 13b , 14a, 14b, we see the failure rates for both high and low nvar($\mathbf{x}$) census tracts in both IL and NC as we vary ε between 2^{-7} and 8. We see that the failure rate is about 40% for any value of ε in low nvar($\mathbf{x}$) and is on average less than 5% for high nvar($\mathbf{x}$) tracts. In Figure 15, we show the failure rate for `NoisyStats` when evaluated on data from all tracts in IL. The results are averaged over 100 trials. We see that the failure rate for $\varepsilon = 8$ is 0% for the majority of tracts. For tracts with small nvar($\mathbf{x}$), the rate of failure is at most 12%. Thus, we can conclude that the failure rate approaches 0 as we increase either ε or nvar($\mathbf{x}$).

D DPEXPTHEILSEN AND DPWIDETHEILSEN

D.1 Privacy Proofs for `DPExpTheilSen` and `DPWideTheilSen`

Lemma 20. *Let T be the following randomized algorithm. For dataset $d = (x_i, y_i)_{i=1}^n$, let $K_n(d)$ be the complete graph on the n datapoints, where edges denote points paired together to compute estimates in Theil-Sen. Then $K_n(d)$ can be decomposed into $n-1$ matchings, $\Sigma_1, \ldots, \Sigma_{n-1}$. Suppose $T(d)$ samples k matchings without replacement from the $n-1$ matchings, and computes the corresponding pairwise estimates (up to $kn/2$ estimates). Then T is a k-Lipschitz randomized transformation.*

Proof. Let $\mathbf{z} = T(d)$ and $\mathbf{z}' = T(d')$ denote the multi-sets of estimates that result from applying T to datasets d and d', respectively. We can define a coupling $\mathbf{z}_c$ and $\mathbf{z}'_c$ of $\mathbf{z}$ and $\mathbf{z}'$. First, use k matchings sampled randomly without replacement from $K_n(d)$, $\Sigma_1, \ldots, \Sigma_k$, to compute the multi-set of estimates $\mathbf{z}_c = \{z_{j,l}^{(p_{xnew})} : (x_j, x_l) \in \Sigma_1 \cup \ldots \cup \Sigma_k\}\}$. Now, use the corresponding k matchings from $K_n(d')$ to compute a multi-set of estimates $\mathbf{z}'_c = \{z_{j,l}^{(p_{xnew})} : (x'_j, x'_l) \in \Sigma_1 \cup \ldots \cup \Sigma_k\}$. This is a valid coupling because the k matchings are sampled randomly without replacement from the complete graphs $K_n(d)$ and $K_n(d')$, respectively, matching the marginal distributions of $\mathbf{z}$ and $\mathbf{z}'$.

Notice that every datapoint x_j is used to compute exactly k estimates in $\mathbf{z}_c$. Therefore, for every datapoint at which d and d' differ, $\mathbf{z}_c$ and $\mathbf{z}'_c$ differ by at most k estimates. Therefore, by the triangle inequality, we are done. □

Proof of Lemma 5. If DPmed($z^{(p25)}, \varepsilon, (n, k,$ hyperparameters)) = $\mathcal{M}(z^{(p25)}$, hyperparameters)) then Algorithm 2 is a composition of two algorithms, $\mathcal{M} \circ T$, where by Lemma 20, T is a k-Lipschitz randomized transformation, and $\mathcal{M}$ is $(\varepsilon/k, 0)$-DP. By the Lipschitz composition lemma (Lemma 18), Algorithm 2 (`DPTheilSenkMatch`) is $(\varepsilon, 0)$-DP. □

Proofs of Lemmas 7 and 8. The privacy of `DPExpTheilSenkMatch` and `DPWideTheilSenkMatch` follows directly from Theorem 13 and Lemma 5. □

D.2 Sensitivity of Hyperparameter

Choosing optimal hyperparameters is beyond the scope of this work. However, in this section we present some preliminary work exploring the behavior of `DPWideTheilSen` with respect to the choice of θ. In particular, we consider the question of how robust this algorithm is to the setting of the hyperparameter. Figure 16 shows the performance as a function the widening parameter θ on synthetic (Gaussian) data. Note that in each graph both axes are on a log-scale so we see very large variation in the quality depending on the choice of hyperparameter.

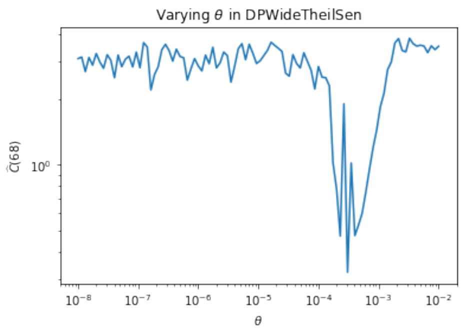


Figure 16: Experimental results exploring the sensitivity of the hyperparameter choices for `DPWideTheilSen`. For each dataset $n = 40$, and n datapoints are generated as $x_i \sim \mathcal{N}(0, \sigma^2)$, $y_i = 0.5 * x_i + 0.5 + \mathcal{N}(0, \tau^2)$. The parameters of the data are fixed at $\sigma = 10^{-3}$ and $\tau = 10^{-4}$. The datapoints are then truncated so they belong between 0 and 1. Note that both axes are on a log scale.

D.3 Pseudo-code for `DPExpTheilSen`

In Algorithm 3, we give an efficient method for implementation of the DP median algorithm used as subroutine in `DPExpTheilSen`, the exponential mechanism for computing medians. To sample efficiently from this distribution, we implement a two-step algorithm following [13]: first, we sample an interval according to the exponential mechanism, and then we will sample an output uniformly at random from that interval. To efficiently sample from the exponential mechanism, we use the fact that sampling from the exponential mechanism is equivalent to choosing the value with maximum utility score after i.i.d. Gumbel-distributed noise has been added to the utility scores [2, 18].

Algorithm 3: Exponential Mechanism for Median: $(\varepsilon/k, 0)$-DP Algorithm

```
Data: $\mathbf{z}$
Privacy params: $\varepsilon$
Input: $n, k, r_l, r_u$
$\varepsilon = \varepsilon/k$
Sort $\mathbf{z}$ in increasing order
Clip $\mathbf{z}$ to the range $[r_l, r_u]$
Insert $r_l$ and $r_u$ into $\mathbf{z}$ and set $n = n + 2$
Set maxNoisyScore $= -\infty$
Set argMaxNoisyScore $= -1$
for $i \in [1, n)$ do
    logIntervalLength $= \log(\mathbf{z}[i] - \mathbf{z}[i-1])$
    distFromMedian $= \lceil |i - \frac{n}{2}| \rceil$
    score $=$ logIntervalLength $- \frac{\varepsilon}{2} \cdot$ distFromMedian
    $N \sim$ Gumbel$(0, 1)$
    noisyScore $=$ score $+ N$
    if noisyScore > maxNoisyScore then
        maxNoisyScore $=$ noisyScore
        argMaxNoisyScore $= i$
left $= \mathbf{z}$[argMaxNoisyScore-1]
right $= \mathbf{z}$[argMaxNoisyScore]
Sample $\tilde{m} \sim$ Unif [left, right]
return $\tilde{m}$
```

D.4 Pseudo-code for DPWideTheilSen

The pseudo-code for DPWideTheilSen is given in Algorithm 4. It is a small variant on Algorithm 3.

E DPSSTHEILSEN

Suppose we are given a dataset $(\mathbf{x}, \mathbf{y})$. Consider a neighboring dataset $(\mathbf{x}', \mathbf{y}')$ that differs from the original dataset in exactly one row. Let $\mathbf{z}$ be the set of point estimates (e.g. the $p25$ or $p75$ point estimates) induced by the dataset $(\mathbf{x}, \mathbf{y})$, and let $\mathbf{z}'$ be the set of point estimates induced by dataset $(\mathbf{x}', \mathbf{y}')$ by Theil-Sen. Formally, for $N = kn/2$, we let $\mathcal{Z}_k : [0,1]^n \times [0,1]^n \to \mathbb{R}^N$ denote the function that transforms a set of point coordinates into estimates for each pair of points. Then $\mathbf{z} = \mathcal{Z}(\mathbf{x}, \mathbf{y})$, $\mathbf{z}' = \mathcal{Z}(\mathbf{x}', \mathbf{y}')$. Notice that changing one datapoint in $(\mathbf{x}, \mathbf{y})$ changes at most k of the point estimates in $\mathbf{z}$. Assume that both $\mathbf{z}$ and $\mathbf{z}'$ are in sorted order. Recall the definition of $S^k_{\text{med}\circ\mathcal{Z},t}((\mathbf{x}, \mathbf{y}))$:

$$S^k_{\text{med}\circ\mathcal{Z}_k,t}((\mathbf{x}, \mathbf{y})) = \max\Big\{z_{m+k} - z_m, z_m - z_{m-k}, \max_{l=1,\ldots,n} \max_{s=0,\cdots,k(l+1)} e^{-lt}(z_{m+s} - z_{m-(k(l+1)+s})\Big\},$$

Let

$$LS^k_{\text{med}}(\mathbf{z}) = \max_{\mathbf{z}' \in \mathbb{R}^N, \text{Ham}(\mathbf{z},\mathbf{z}') \le k} |\text{med}(\mathbf{z}) - \text{med}(\mathbf{z}')|$$

Algorithm 4: θ-Widened Exponential Mechanism for Median: $(\varepsilon/k, 0)$-DP Algorithm

```
Data: $\mathbf{z}$
Privacy params: $\varepsilon$
Input: $n, k, \theta, r_l, r_u$
$\varepsilon = \varepsilon/k$
Sort $\mathbf{z}$ in increasing order
Clip $\mathbf{z}$ to the range $[r_l, r_u]$
if $n$ is even then
    Insert $m$, the true median, into $\mathbf{z}$
    Set $n = n + 1$
for $i \in [0, \lfloor \frac{n}{2} \rfloor]$ do
    $z[i] = \max(z_l, z[i] - \theta)$
    $z[n-i-1] = \min(z_u, z[i] + \theta)$
Insert $z_l$ and $z_b$ into $\mathbf{z}$ and set $n = n + 2$
Set maxNoisyScore $= -\infty$
Set argMaxNoisyScore $= -1$
for $i \in [1, n)$ do
    logIntervalLength $= \log(\mathbf{z}[i] - \mathbf{z}[i-1])$
    distFromMedian $= \lceil |i - \frac{n}{2}| \rceil$
    score $=$ logIntervalLength $- \frac{\varepsilon}{2} \cdot$ distFromMedian
    $N \sim$ Gumbel$(0, 1)$
    noisyScore $=$ score $+ N$
    if noisyScore > maxNoisyScore then
        maxNoisyScore $=$ noisyScore
        argMaxNoisyScore $= i$
left $= \mathbf{z}$[argMaxNoisyScore-1]
right $= \mathbf{z}$[argMaxNoisyScore]
Sample $\tilde{m} \sim$ Unif [left, right]
return $\tilde{m}$
```

be distance k local sensitivity of the dataset $\mathbf{z}$ with respect to the median. In order to prove that $S^k_{\text{med}\circ\mathcal{Z}_k,t}((\mathbf{x}, \mathbf{y}))$ is a t-smooth upper bound on $LS_{\text{med}\circ\mathcal{Z}_k}$, we will use the observation that

$$LS_{\text{med}\circ\mathcal{Z}_k}(\mathbf{x}, \mathbf{y}) \le LS^k_{\text{med}}(\mathbf{z}).$$

Now Figure 17 outlines the maximal changes we can make to $\mathbf{z}$. For $l \ge 1$ and any interval of $lk + k + 1$ points containing the median, we can move the median to one side of the interval by moving kl points, and to the other side by moving an additional l points. Therefore, for $l \ge 1$,

$$\max_{\mathbf{z}':d(\mathbf{z},\mathbf{z}') \le lk} LS_{\text{med}}(\mathbf{z}') = \max_{s=0,\cdots,lk+k} \{z_{m+s} - z_{m-(lk+k)+s}\} \quad (4)$$

so

$$S^k_{\text{med},t}(\mathbf{z}) = \max_{l=0,\ldots,n} e^{-lt} \max_{\mathbf{z}':d(\mathbf{z},\mathbf{z}') \le lk} LS^k_{\text{med}}(\mathbf{z}').$$

Proof of Lemma 10. We need to show that $S^k_{\text{med}\circ\mathcal{Z}_k,t}((\mathbf{x}, \mathbf{y}))$ is lower bounded by the local sensitivity and that for any dataset $(\mathbf{x}', \mathbf{y}')$ such that $d((\mathbf{x}, \mathbf{y}), (\mathbf{x}', \mathbf{y}')) \le l$, we have $S^k_{\text{med}\circ\mathcal{Z}_k,t}((\mathbf{x}, \mathbf{y})) \le e^{tl} S^k_{\text{med}\circ\mathcal{Z}_k,t}((\mathbf{x}', \mathbf{y}')).$

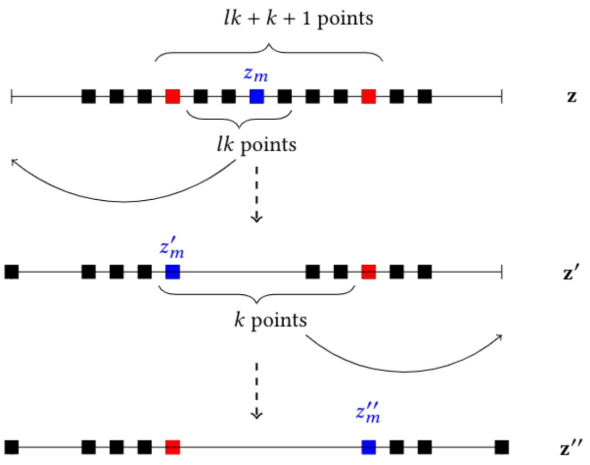


Figure 17: A brief proof by pictures of Equation 4.

By definition of $S^k_{\text{med},t}$, we see that $S^k_{\text{med}\circ\mathcal{Z}_k,t}((\mathbf{x},\mathbf{y})) \geq LS^k_{\text{med}}$ (e.g. when $l = 0$ in the formula for $S^k_{\text{med},t}$). Next, we see that

$$S^k_{\text{med},t}(\mathbf{z}) = \max_{l=0,\dots,n} e^{-lt} \max_{\mathbf{z}':d(\mathbf{z},\mathbf{z}')\leq lk} LS^k_{\text{med}}(\mathbf{z}') \tag{5}$$

$$\leq e^t \cdot \max_{l=1,\dots,n} e^{-lt} \max_{\mathbf{z}'':d(\mathbf{z}',\mathbf{z}'')\leq lk} LS^k_{\text{med}}(\mathbf{z}'') \tag{6}$$

$$\leq e^t \cdot S^k_{\text{med},t}(\mathbf{z}'), \tag{7}$$

which completes our proof. □

Lemma 21. *Let $M(\boldsymbol{x}) = median(\boldsymbol{x}) + \frac{1}{s}S^k_{med,t}(\boldsymbol{x}) \cdot N$, where N, s and t are computed according to Algorithm 5. Then, M is $(\varepsilon, 0)$-DP.*

Proof. Let $D_\infty(P||Q) = \sup_{x\in\text{supp}(Q)} \log\frac{p(x)}{q(x)}$ denote the max-divergence for distributions P and Q. Let N be a random variable sampled from StudentsT(d), where $d > 0$ is the degrees of freedom. From Theorem 31 in [7], we have that for $s, t > 0$,

$$\left.\begin{array}{l} D_\infty(N||e^tN+s) \\ D_\infty(e^tN+s||N) \end{array}\right\} \leq |t|(d+1) + |s| \cdot \frac{d+1}{2\sqrt{d}}$$

The parameters s and t correspond to the translation (shifting) and dilation (scaling) of the StudentsT(d) distribution.

Setting $s = 2\sqrt{d}\left(\frac{\varepsilon'-|t|(d+1)}{d+1}\right)$ as in Algorithm 5, we have that for $|t|(d+1) < \varepsilon'$,

$$\begin{cases} D_\infty(N||e^tN+s) \\ D_\infty(e^tN+s||N) \end{cases} \leq \varepsilon \tag{8}$$

If Equation 8 is satisfied, then by Theorem 46 in [7], the mechanism in Algorithm 5, $M(\mathbf{z}) = \text{median}(\mathbf{z}) + \frac{1}{s}S^t_{\text{median}(\cdot)}(\mathbf{z}) + N$, is $(\varepsilon, 0)$-DP. □

E.1 Sensitivity of Hyperparameters

In Algorithm 5 we set the smoothing parameter to be a specific function of ϵ and d: $t = \frac{\varepsilon}{2(d+1)}$ and $s = \frac{\epsilon\sqrt{d}}{d+1}$. There were other choices

Algorithm 5: Smooth Sensitivity Student's T Noise Addition for Median: $(\varepsilon, 0)$-DP Algorithm

```
Data: z, $\{(x_i, y_i)\}_{i=1}^n \in ([0,1]\times[0,1])^n$
Privacy params: $\varepsilon$
Hyperparams: $k, n, r_l, r_u, d$
Set $t = \frac{\varepsilon}{2(d+1)}$ and $s = \frac{\varepsilon\sqrt{d}}{d+1}$
$S_{\text{median}} = S^k_{\text{med},t}((\mathbf{x},\mathbf{y}))$
Sample $N \sim$ Student's T($d$)
Set $\widetilde{m} = \text{median}(\mathbf{z}) + \frac{1}{s}\cdot S_{\text{median}}\cdot N$
return $\widetilde{m}$
```

for these parameters. For any $\beta \in [0, 1]$, the $(\epsilon, 0)$-DP guarantee is preserved if we set

$$t = \frac{\epsilon\beta}{d+1} \quad \text{and} \quad s = 2\sqrt{d}\left(\frac{\epsilon - t(d+1)}{d+1}\right).$$

Algorithm 5 corresponds to setting $\beta = 1/2$. Increasing β increases t, which results in $S^k_{\text{med}\circ\mathcal{Z}_k}((\mathbf{x},\mathbf{y}))$ decreasing. However, if increasing β also decreases s. In Figure 18 we explore the performance of DPSSTheilSen as a function of β on synthetic (Gaussian) data. Note that the performance doesn't seem too sensitive to the choice of β and $\beta = 0.5$ is a good choice on this data.

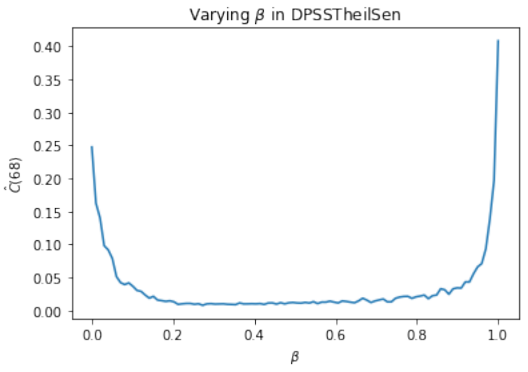


Figure 18: Experimental results exploring the sensitivity of the hyperparameter choices for DPSSTheilSen. For each dataset $n = 30$, and n datapoints are generated as $x_i \sim \mathcal{N}(0.5, \sigma^2)$, $y_i = 0.5 * x_i + 0.2 + \mathcal{N}(0, \tau^2)$. The parameters of the data are fixed at $\sigma = 0.01$ and $\tau = 0.005$. The datapoints are then truncated so they belong between 0 and 1. Note that both axes are on a log scale.

F DPGRADDESCENT

There are three main versions of DPGradDescent we consider:

(1) DPGDPure: $(\varepsilon, 0)$-DP;
(2) DPGDApprox: (ε, δ)-DP; and
(3) DPGDzCDP: $(\varepsilon^2/2)$-zCDP.

Algorithm 6 is the specification of a $(\varepsilon, 0)$-DP algorithm and Algorithm 7 is a ρ-zCDP algorithm, from which we can obtain a (ε, δ)-DP algorithm and a $(\varepsilon^2/2)$-zCDP algorithm. As with traditional gradient descent, there are several choices that have been made in designing this algorithm: the step size, the batch size for the gradients, how many of the estimates are averaged to make our final estimate, how the privacy budget is distributed. We have included this pseudo-code for completeness to show the choices that were made in our experiments. We do not claim to have extensively explored the suite of parameter choices, and it is possible that a better choice of these parameters would result in a better performing algorithm. Differentially private gradient descent has received a lot of attention in the literature. For a more in-depth discussion of DP gradient descent see [4].

Algorithm 6: `DPGDPure`: $(\varepsilon, 0)$-DP Algorithm

```
Data: $\{(x_i, y_i)\}_{i=1}^n \in ([0,1] \times [0,1])^n$
Privacy params: $\varepsilon$
Hyperparams: $n, T, \tau, \tilde{p}_{25}^0, \tilde{p}_{75}^0$
for $t = 0 : T-1$ do
    $\varepsilon_t = \varepsilon / T$
    for $i = 1 : n$ do
        $\tilde{y}_{i,t} = 2(\tilde{p}_{25}^t * (3/4 - x_i) + \tilde{p}_{75}^t (x_i - 1/4))$
        $\Delta_{i,t} = \begin{pmatrix} [2(y_i - \tilde{y})(3/4 - x_i)]_{-\tau}^{\tau}, \\ [2(y_i - \tilde{y})(x_i - 1/4)]_{-\tau}^{\tau} \end{pmatrix}$
    $\Delta_t = \sum_{i=1}^n \Delta_{i,t} + \mathrm{Lap}_2(0, 4\tau/\varepsilon_t)$
    $\gamma_t = \frac{1}{\sqrt{\sum_{l=0}^t \Delta_l^2}}$
    $[\tilde{p}_{25}^{t+1}, \tilde{p}_{75}^{t+1}] = [\tilde{p}_{25}^t, \tilde{p}_{75}^t] - \gamma_t * \Delta_t$
return $\frac{2}{T} \sum_{t=T/2}^{T-1} [\tilde{p}_{25}^t, \tilde{p}_{75}^t]$
```

Algorithm 7: `DPGDzCDP`: ρ-zCDP Algorithm

```
Data: $\{(x_i, y_i)\}_{i=1}^n \in ([0,1] \times [0,1])^n$
Privacy params: $\rho$
Hyperparams: $n, T, \tau, \tilde{p}_{25}^0, \tilde{p}_{75}^0$
for $t = 0 : T-1$ do
    $\rho_t = \rho / T$
    for $i = 1 : n$ do
        $\tilde{y}_{i,t} = 2(\tilde{p}_{25}^t * (3/4 - x_i) + \tilde{p}_{75}^t (x_i - 1/4))$
        $\Delta_{i,t} = \begin{pmatrix} [2(y_i - \tilde{y})(3/4 - x_i)]_{-\tau}^{\tau}, \\ [2(y_i - \tilde{y})(x_i - 1/4)]_{-\tau}^{\tau} \end{pmatrix}$
    $\Delta_t = \sum_{i=1}^n \Delta_{i,t} + \mathcal{N}_2\left(0, (2\tau/\sqrt{\rho}_t)^2\right)$
    $\gamma_t = \frac{1}{\sqrt{\sum_{l=0}^t \Delta_l^2}}$
    $[\tilde{p}_{25}^{t+1}, \tilde{p}_{75}^{t+1}] = [\tilde{p}_{25}^t, \tilde{p}_{75}^t] - \gamma_t * \Delta_t$
return $\frac{2}{T} \sum_{t=T/2}^{T-1} [\tilde{p}_{25}^t, \tilde{p}_{75}^t]$
```

Lemma 22. *For any $\rho > 0$, Algorithm 7 is ρ-zCDP.*

Proof. By composition properties of zCDP, it suffices to show that $\Delta_t = \sum_{i=1}^n \Delta_{i,t} + \mathcal{N}_2\left(0, (2\tau/\sqrt{\rho}_t)^2\right)$ is ρ_t-zCDP where $\mathcal{N}_2(0, s^2)$ represents two (independent) draws from a Normal distribution with standard deviation s.

The L_2-sensitivity of $\sum_{i=1}^n \Delta_{i,t}$ is at most $2\sqrt{2}\tau$ and the standard deviation of the Gaussian distribution from which noise is added is $2\tau/\sqrt{\rho_t}$. Then by Proposition 1.6 in [6], the procedure to compute Δ_t is ρ_t-zCDP. □

Lemma 23. *For any $\delta \in (0,1]$ and any $\rho > 0$, Algorithm 7 is (ε, δ)-DP where*

$$\varepsilon = \rho + \sqrt{4\rho \log\left(\frac{\sqrt{\pi\rho}}{\delta}\right)}.$$

Proof. Follows from Lemma 3.6 in [6]. □

Lemma 24. *For any $\varepsilon > 0$, Algorithm 6 is $(\varepsilon, 0)$-DP.*

Proof. By basic composition, it suffices to show that $\Delta_t = \sum_{i=1}^n \Delta_{i,t} + \mathrm{Lap}_2(0, 4\tau/\varepsilon_t)$ is $(\varepsilon_t, 0)$-DP where $\mathrm{Lap}_2(0, s)$ represents two (independent) draws from a Laplace distribution with scale s. This holds since the L_1-sensitivity of $\sum_{i=1}^n \Delta_{i,t}$ is at most 4τ. □