Improved Differentially Private Regression via Gradient Boosting

Shuai Tang[1], Sergul Aydore[1], Michael Kearns[1, 2], Saeyoung Rho[3], Aaron Roth[1, 2], Yichen Wang[1], Yu-Xiang Wang[1, 4], and Zhiwei Steven Wu[1, 5]

[1]Amazon AWS AI/ML
[2]University of Pennsylvania
[3]Columbia University
[4]University of California, Santa Barbara
[5]Carnegie Mellon University

***Abstract*—We revisit the problem of differentially private squared error linear regression. We observe that existing state-of-the-art methods are sensitive to the choice of hyperparameters — including the "clipping threshold" that cannot be set optimally in a data-independent way. We give a new algorithm for private linear regression based on gradient boosting. We show that our method consistently improves over the previous state of the art when the clipping threshold is taken to be fixed without knowledge of the data, rather than optimized in a non-private way — and that even when we optimize the hyperparameters of competitor algorithms non-privately, our algorithm is no worse and often better. Additional experiments also that our algorithm is also more robust to outliers.**

***Index Terms*—differential privacy, linear regression, gradient boosting**

I. Introduction

Squared error linear regression is a basic, foundational method in statistics and machine learning. Absent other constraints, it has an optimal closed-form solution. A consequence of this is that linear regression parameters have a deterministic relationship with the data they are fitting, which can leak private information. As a result, there is a substantial body of work aiming to approximate the solution to least squares linear regression with the protections of differential privacy [1]–[7].

We highlight the AdaSSP ("Adaptive Sufficient Statistics Perturbation") algorithm [4] which obtains state-of-the-art theoretical and practical performance when the maximum norm of the features and labels are known—these bounds are used to scale the noise added for privacy. When a data-independent bound on the magnitude of the data is not known, in order to promise differential privacy, they must be clipped at some data-independent threshold, which can substantially harm performance. In this work, we give a new algorithm for private linear regression that substantially mitigates this issue and leads to improved accuracy across a range of datasets and clipping thresholds.

Our approach is both conceptually and computationally simple: we apply gradient boosting [8], using a linear model as the base learner, and to incorporate privacy guarantees, at each boosting round, the linear model is solved using AdaSSP. When applied to a squared error objective, gradient boosting is exceedingly simple: it maintains a linear combination of regression models, repeatedly fitting a new regression model to the *residuals* of the current model, and then adding the new model to the linear combination. Absent privacy constraints, gradient boosting for linear regression does not improve performance, because linear models are closed under linear combinations, and squared error regression can be optimally solved over the set of all linear models in closed form. Nevertheless, in the presence of privacy constraints and in the absence of knowledge of the data scale (so that we must use a data independent clipping threshold), we show in an extensive set of experiments that gradient BoostedAdaSSP substantially improves on the performance of AdaSSP alone. Moreover, we show that our BoosedAdaSSP algorithm outperforms other competitive differentially private solutions to linear regression in different conditions, including gradient descent on the squared loss objective, and interestingly performs better than a tree-based private boosting algorithm. We also show that our algorithm is less sensitive to hyperparameter selection.

Due to the boosting nature of our algorithm, it leverages many benefits of boosting algorithms, and one of them is robustness to outliers. On a simulated dataset, we consider three corruption approaches that generate outliers, including label, feature and model corruption,and we show that our algorithm is indeed more robust to outliers than the original AdaSSP, and also more robust to a strong competitor in many scenarios.

In addition to our empirical results, we provide stylized theoretical explanations. In the zero-dimensional case, AdaSSP reduces to computing the empirical mean of the clipped data, and aggressive clipping thresholds can cause the bias of empirical mean to be arbitrarily large. In this setting, gradient boosting with AdaSSP as a base learner corresponds to iteratively updating an estimator of the mean by the clipped empirical residuals , i.e. the empirical mean of the difference between the current mean estimate and the data. In Section VI, we show that, for Gaussian data, the boosting method converges to the true mean for *any* non-zero clipping threshold.

The intuition behind this improvement of boosting over the one-shot empirical mean is that, even clipped estimates of the mean are directionally correct, which serves to further de-bias the current estimate and reduce the negative effect of aggressive clipping. The convergence of our boosted algorithm under arbitrary clipping provides a significant improvement over AdaSSP, especially when the clipping bound must be independent to the data.

Finally, we show that BoostedAdaSSP can sometimes outperform differentially private boosted trees [9] as well, a phenomenon that we do not observe absent privacy. This contributes to an important conceptual message: that the best learning algorithms under the constraint of differential privacy are not necessarily "privatized" versions of the best learning algorithms absent privacy—differential privacy rewards algorithmic simplicity.

A. Additional Related Work

Because of its fundamental importance, linear regression has been the focus of a great deal of attention in differential privacy [1]–[7], [10], using techniques including private gradient descent [11], [12], output and objective perturbation [13], and perturbation of sufficient statistics [14]. As already mentioned, the AdaSSP (a variant of the sufficient statistic perturbation approach) [4] has stood out as a method obtaining both optimal theoretical bounds and strong empirical performance — both under the assumption that the magnitude of the data is known.

[7] have previously noted that AdaSSP can perform poorly when the data magnitude is unknown and clipping bounds must be chosen in data-independent ways. They also give a method — TukeyEM [7] — aiming to remove these problematic hyperparameters for linear regression. TukeyEM privately aggregates multiple non-private linear regressors learned on disjoint subsets of the training set. The private aggregate uses the approximate Tukey depth and removes the risk of potential privacy leaks in choosing hyperparameters. However, because each model is trained on a different partition of the data, as [7] note, TukeyEM performs well when the number of samples is roughly $1,000$ times larger than the dimension of the data. We include a comparison to both TukeyEM and AdaSSP in our experimental results.

Another line of work has studied differentially private gradient boosting methods, generally using a weak learner class of classification and regression trees (CARTs) [15], [16]. [9] gives a particularly effective variant called DP-EBM, which we compare to in our experiments.

There is a line of work that aims to privately optimize hyperparameters (e.g. [17]–[19]) — we do not directly compare to these approaches, but our experiments show that our algorithm dominates comparison methods even when their hyperparameters are optimized non-privately.

II. Preliminaries

We study the standard squared error linear regression problem. Given a joint distribution $\mathcal{D}$ over p dimensional features $x \in \mathbb{R}^p$ and real-valued labels $y \in \mathbb{R}$. Our goal is to learn a parameter vector $\theta \in \mathbb{R}^p$ to minimize squared error:

$$\mathcal{L}(\theta, \mathcal{D}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[(\langle\theta, x\rangle - y)^2]. \tag{1}$$

In order to protect privacy of individuals in the training data when the learnt parameter vector θ is released, we adopt the notion of Differential Privacy.

A. Differential Privacy (DP)

Differential privacy is a strong formal notion of individual privacy. DP ensures that, for a randomized algorithm, when two neighboring datasets that differ in one data point are presented, the two outputs are indistinguishable, within some probability margin defined using ϵ and $\delta \in [0, 1)$.

Definition II.1 (Differential Privacy [20])**.** A randomized algorithm $\mathcal{M}$ with domain $\mathcal{D}$ is (ϵ, δ)-differentially private for all $\mathcal{S} \subseteq \text{Range}(\mathcal{M})$ and for all pairs of neighboring databases $D, D' \in \mathcal{D}$,

$$\Pr[\mathcal{M}(D) \in \mathcal{S}] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in \mathcal{S}] + \delta, \tag{2}$$

where the probability space is over the randomness of the mechanism $\mathcal{M}$.

A refinement of differential privacy, a single-parameter privacy definition (Gaussian differential privacy, GDP) was later proposed [21]. In this work, we use GDP in order to achieve better privacy bounds. We present several key results in [21] that we use in our privacy analysis.

Definition II.2 (ℓ_2-sensitivity)**.** The ℓ_2-sensitivity of a statistic m over the domain of dataset D is $\Delta(m) = \sup_{D,D'} \|m(D) - m(D')\|_2$, where $\|\cdot\|_2$ is the vector ℓ_2-norm, and the supremum is over all neighboring datasets.

Theorem II.3 (Gaussian Mechanism, Theorem 2.7 from of [21])**.** *Define a randomized algorithm GM that operates on a statistic m as $GM(x, \mu) = m(x) + \eta$, where $\eta \sim \mathcal{N}(0, \Delta(m)^2/\mu^2)$ and $\Delta(m)$ is the ℓ_2-sensitivity of the statistics m. Then, GM is μ-GDP.*

For n GDP mechanisms with privacy parameters $\mu_1, \cdots, \mu_n$,the following composition theorem holds:

Corollary II.4 (Composition of GDP, Corollary 3.3 of [21])**.** *The n-fold composition of μ_i-GDP mechanisms is $\sqrt{\mu_1^2 + \cdots + \mu_n^2}$-GDP.*

There is a tight relationship between μ-GDP and (ϵ, δ)-DP that allows us to perform our analysis using GDP, and state our results in terms of (ϵ, δ)-DP.

Corollary II.5 (Conversion between GDP and DP, Corollary 2.13 of [21])**.** *A mechanism is μ-GDP if and only if it is $(\epsilon, \delta(\epsilon))$-DP for all $\epsilon \geq 0$, such that*

$$\delta(\epsilon) = \Phi\left(-\frac{\epsilon}{\mu} + \frac{\mu}{2}\right) - e^{\epsilon}\Phi\left(-\frac{\epsilon}{\mu} - \frac{\mu}{2}\right) \tag{3}$$

where Φ denotes the standard Gaussian CDF.

III. Improved AdaSSP via Gradient Boosting

Our algorithm for private linear regression uses gradient boosting with AdaSSP as a weak learner.

A. Gradient Boosting

For regression tasks, we assume that we have a dataset $D = \{x_i, y_i\}_{i=1}^n$, where $x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$, $\forall i \in [n]$. Let T be the number of boosting rounds, and f_t be the model obtained at iteration $t \in [T]$. Since our base learner is linear and the objective is the squared loss, at the t-th round, the objective of a gradient boosting algorithm is to obtain:

$$\theta_t = \arg\min_\theta \sum_{i=1}^n (y_i - (\sum_{k=1}^{t-1} \theta_k^\top x_i + \theta^\top x_i))^2 \quad (4)$$

$$= \arg\min_\theta \sum_{i=1}^n \left(g_{i,t} - \theta^\top x_i\right)^2, \quad (5)$$

where $g_{i,t} = y_i - \sum_{k=1}^{t-1} \theta_k^\top x_i$ is the steepest gradient of the objective function w.r.t. the ensemble predictions made by previous rounds. Therefore, each gradient boosting round is solving a squared error linear regression problem where the features are data, and the labels are gradients. The model update at t-th round is simply $\hat{\theta} = \hat{\theta} + \theta_t$, and the final model is $\hat{\theta} = \sum_{k=1}^T \theta_k$.

Since the update preserves the linearity of the model, and squared error regression can be solved optimally over linear models. Absent privacy, gradient boosting cannot improve the error of linear regression in the standard setting. Nevertheless, when we replace exact linear regression with differentially private approximations, the situation changes.

B. Private Ridge Regression as a Base Learner

Let $X \in \mathbb{R}^{n \times p}$ be the matrix with x_i's in each row and $g_t \in \mathbb{R}^n$ be a vector containing gradients of training samples at t (i.e., $g_{i,t}$). Absent privacy, there exists a closed-form soluion to Eq. 5, and it is

$$\theta_t = (X^\top X)^{-1} X^\top g_t, \quad (6)$$

To provide differential privacy guarantees, AdaSSP (Algorithm 2 of [4]) is applied to learn a private linear model at each round. It also requires us to adjust our solution at each round from OLS to Ridge Regression as follows:

$$\theta_t = (X^\top X + \lambda I)^{-1} X^\top g_t, \quad (7)$$

where λ controls the strength of regularization, and $I \in \mathbb{R}^{p \times p}$ is the identity matrix.

Let $\mathcal{X}$ and $\mathcal{Y}$ be the domain of our features and labels, respectively. We define bounds on the data domain $||\mathcal{X}|| = \sup_{x \in \mathcal{X}} ||x||$ and $||\mathcal{Y}|| = \sup_{y \in \mathcal{Y}} ||y||$. Given as input privacy parameters ϵ and δ, and bounds on the data scale $||\mathcal{X}||$ and $||\mathcal{Y}||$ for x_i and $g_{i,t}$, AdaSSP chooses a noise scale to obtain μ-GDP for the appropriate value of μ, and adds calibrated Gaussian noise to three sufficient statistics: 1) $X^\top X$, 2) $X^\top g_t$, and 3) λ. The adaptive aspect of AdaSSP comes from the fact that λ is chosen based on $X^\top X$, therefore, we also need to allocate privacy budget for computing $\hat{\lambda}$. Details of the AdaSSP algorithm for learning one ridge regressor are deferred to Appendix B.

Algorithm 1 BoostedAdaSSP

Input Dataset $D = \{X, y\}$, privacy parameters ϵ, δ, split ratio a, b, c, and clipping threshold τ
Initialize $\theta = 0$
Find μ such that μ-GDP satisfies (ϵ, δ)-DP.
Calibrate μ_1, μ_2, μ_3 such that $\mu_1 : \mu_2 : \mu_3 = a : b : c$ and $\mu = \sqrt{\mu_1^2 + \mu_2^2 + \mu_3^2}$
Clip the norm of samples x_i = clip(x_i, 1), $\forall i \in [n]$
Private release of $\widehat{\lambda} = GM(\lambda_{\min}(X^\top X), \mu_3)$ and $\widehat{X^\top X} = GM(X^\top X, \mu_1)$
Compute $\Gamma = (\widehat{X^\top X} + \widehat{\lambda} I)^{-1}$
for $t \in [T]$ **do**
  $g_t = y - X\theta$ `# negative gradient`
  $g_{i,t} = \text{clip}(g_{i,t}, \tau)$, $\forall i \in [n]$ `# gradient clipping`
  $\widehat{X^\top g_t} = GM(X^\top g_t, \frac{\mu_2}{\sqrt{T}})$ `# private release`
  $\theta_t = \Gamma \widehat{X^\top g_t}$ `# private linear model`
  $\theta \leftarrow \theta + \theta_t$ `# model update`
end for
Return θ
* $\text{clip}(x, \tau) = x \min(1, \tau / ||x||)$
* $GM(X, \mu)$ denotes the Gaussian mechanism that satisfies μ-GDP for releasing noisy X

Let $\widehat{X^\top X} = GM(X^\top X, \mu_1)$, $\widehat{X^\top g_t} = GM(X^\top g_t, \mu_2)$, $\widehat{\lambda} = GM(\lambda_{\min}(X^\top X), \mu_3)$ be the private release of sufficient statistics from a single instantiation of AdaSSP to learn θ_t and GM as defined in Theorem II.3. The final model $\widehat{\theta}$ can be expressed as

$$\widehat{\theta} = \sum_{t=1}^T \widehat{\theta}_t = \left(\widehat{X^\top X} + \widehat{\lambda} I\right)^{-1} \sum_{t=1}^T \widehat{X^\top g_t} \quad (8)$$

Therefore, when running gradient boosting, we only need to release $GM(X^\top X, \mu_1)$ and $GM(\lambda_{\min}(X^\top X), \mu_3)$ once at the beginning of our algorithm, and at each stage, the only additional information we need to release is $GM(X^\top g_t, \mu_2/\sqrt{T})$; this provides a savings over naively repeating AdaSSP (given as Algorithm 2 in the Appendix) for T rounds.

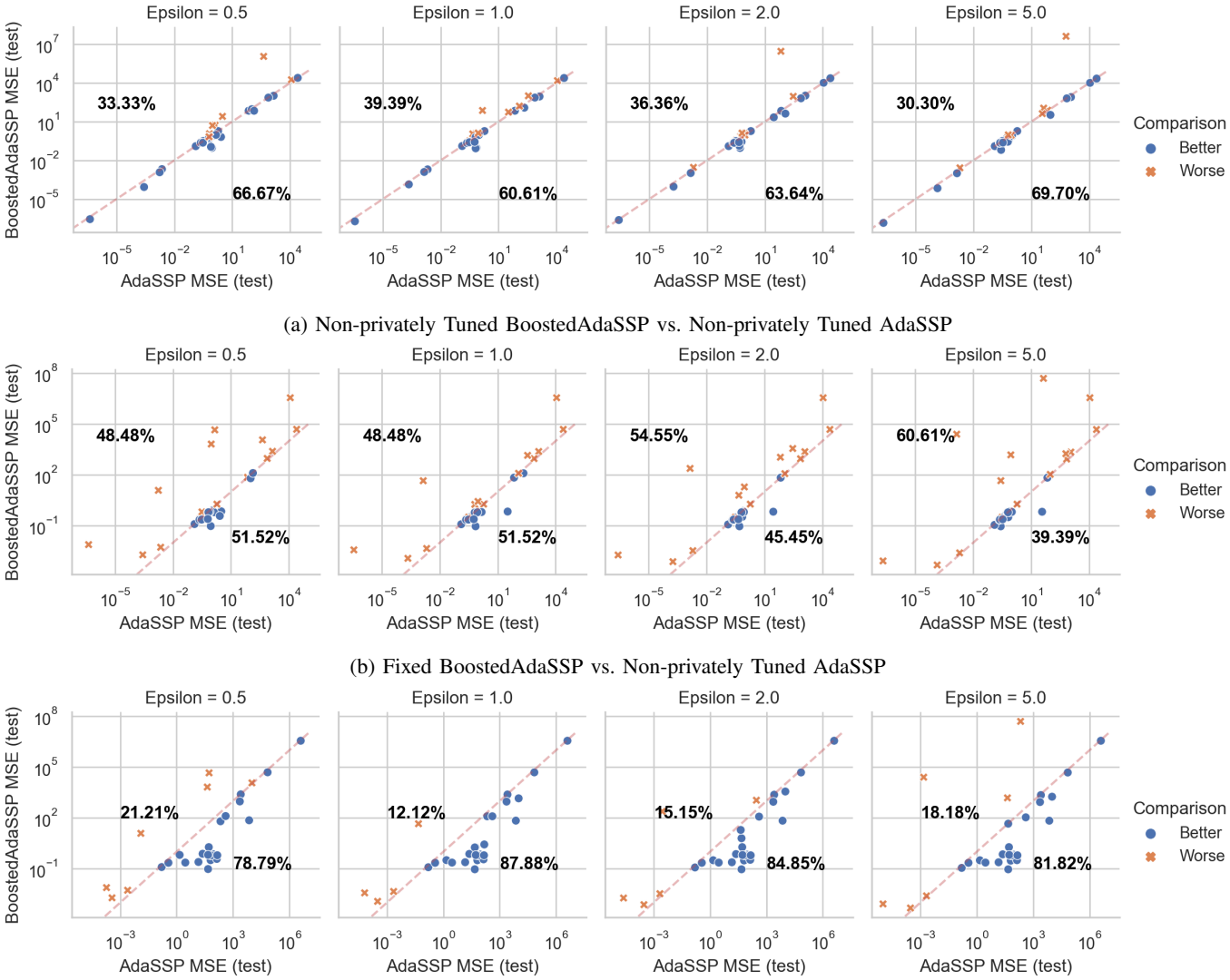
Putting it all together, our final algorithm BoostedAdaSSP is shown in Algorithm 1, and the privacy guarantee is shown in Theorem III.1.

Theorem III.1. *Algorithm 1 satisfies (ϵ, δ)-DP.*

Proof. We use the privacy of the Gaussian mechanism, and the composition theorem stated in corollary II.4, which gives us a GDP bound of: $\sqrt{\mu_1^2 + T\left(\mu_2/\sqrt{T}\right)^2 + \mu_3^2} = \sqrt{\mu_1^2 + \mu_2^2 + \mu_3^2} = \mu$. The conversion from GDP to DP follows from Corollary II.5. □

C. Data-independent Clipping Bounds

As described in [4] and mentioned in [7], the clipping bounds on $\mathcal{X}$ and $\mathcal{Y}$ are taken to be known — but if they are selected



(a) Non-privately Tuned BoostedAdaSSP vs. Non-privately Tuned AdaSSP

(b) Fixed BoostedAdaSSP vs. Non-privately Tuned AdaSSP

(c) Fixed BoostedAdaSSP vs. Fixed AdaSSP

Fig. 1: **BoostedAdaSSP vs. AdaSSP.** "Non-privately Tuned" indicates that hyperparameters of the algorithm are non-privately optimized on each dataset, and "Fixed" indicates that the hyperparameters are fixed and shared across all datasets. Each dataset is shown as a point on the plot, labeled with the error obtained by BoostedAdaSSP (y axis) and AdaSSP (x axis). Points below the diagonal are datasets on which BoostedAdaSSP improves over AdaSSP—the fractions of datasets lying above and below the diagonal are annotated.

as a deterministic function of the data, this would constitute a violation of differential privacy. For $\mathcal{Y}$, the most natural solution is to use a data-independent τ to clip labels and enforce a bound of τ; but as we observe both empirically and theoretically, this introduces a difficult-to-tune hyperparameter that can lead to a substantial degradation in performance. For $\mathcal{X}$, one way to resolve this issue, (as is done in the implementation of AdaSSP [1]) is to normalize each individual data point to have norm 1, but this is not without loss of generality: it fundamentally changes the nature of the regression problem being solved, and so does not always constitute a meaningful solution to the original problem. Instead, we clip the norm of data points so that the maximum norm doesn't exceed a fixed data independent threshold (but might be lower).

IV. Experiments

We selected 33 tabular datasets with single-target regression tasks from OpenML [2] [22] for evaluating and comparing our algorithm to other algorithms. Task details are presented in Table II . The selected tasks include both categorical and numerical features. We assume that the schema of individual

[1] https://github.com/yuxiangw/autodp/blob/master/tutorials/tutorial_AdaSSP_vs_noisyGD.ipynb

[2] https://www.openml.org

tables is public information, and so convert categorical features into one-hot encodings.

We compare our approach with a number of other algorithms. First, we compare to other private linear regression methods: AdaSSP, DP Gradient Descent, and TukeyEM. These represent the leading practical methods (with accompanying code) used for solving linear regression problems. DP Gradient Descent solves the linear regression problem through noisy batch gradient descent with noise calibrated with clipped per-sample gradients, meanwhile, TukeyEM trains nonprivate linear models on disjoint subsets and privately aggregates the learned linear models. Since our algorithm is based on gradient boosting, in addition to algorithms that solve linear regression problems, we also compare to DP-EBM [3], the current state-of-the-art differentially private gradient boosting algorithm, which uses trees as its base learners. Rather than finding the optimal splits for each leave based on the data, DP-EBM uses random splits, which significantly improves the efficacy of the privacy budget. We also include our own implementation of DP-Adaptive-Mini-Batch-Shuffled-SGD (DP-AMBSSGD) [6], as it currently doesn't have publicly available codebase.

As each algorithm has it own hyperparameters (which are often tuned non-privately in reported results), we present three sets of comparisons. 1) First, we compare performance of the algorithms when the hyperparameters are non-privately optimized for each dataset, for each of the algorithms. This provides an (unrealistically) optimistic view of each algorithm's best case performance. 2) Next, we use a fixed set of hyperparameters for our algorithm (BoostedAdaSSP), which remain unchanged from dataset to dataset, while still non-privately optimizing the hyperparameters of each of our comparison partners on a dataset-by-dataset basis. This provides an (unrealistic) best-case comparison for the methods we benchmark against. 3) Finally, we show what we view as the fair comparison, which is when the hyperparameters of our method (BoostedAdaSSP) as well as those of all of our comparison partners are held constant across all of the datasets. For hyperparameter tuning, Optuna [23] is applied. The tuning ranges of hyperparameters, and the fixed hyperparameters for our method are reported in Table I in the Appendix. For each comparison partner, when we fix the parameters, we use parameters recommended in their papers.

Gradient Boosting Improves AdaSSP. When hyperparameters are non-privately tuned for both methods, then the mean squared error is quite similar on most datasets for both methods, but our method (BoostedAdaSSP) obtains lower error on the majority of datasets at all tested privacy levels. When BoostedAdaSSP uses fixed hyperparameters, it remains competitive with AdaSSP even when AdaSSP is non-privately tuned on each dataset. Finally, when both methods use fixed hyperparameters, BoostedAdaSSP has substantially improved error across a majority of datasets at all privacy levels. This indicates a substantial advantage for our method. Comparisons are presented in Fig. 1.

[3] https://github.com/interpretml/interpret

BoostedAdaSSP outperforms DP-Gradient Descent. Gradient descient and BoostedAdaSSP are similar iterative algorithms. But in all comparison settings (including when the hyper-parameters of gradient descent are non-privately optimized on individual datasets, and BoostedAdaSSP uses fixed hyperparameters across all datasets), BoostedAdaSSP substantially outperforms. BoostedAdaSSP can be viewed as gradient descent in function space rather than parameter space, and is able to take advantage of the optimized ridge regression estimator of AdaSSP at each step. Results are in Fig. 2.

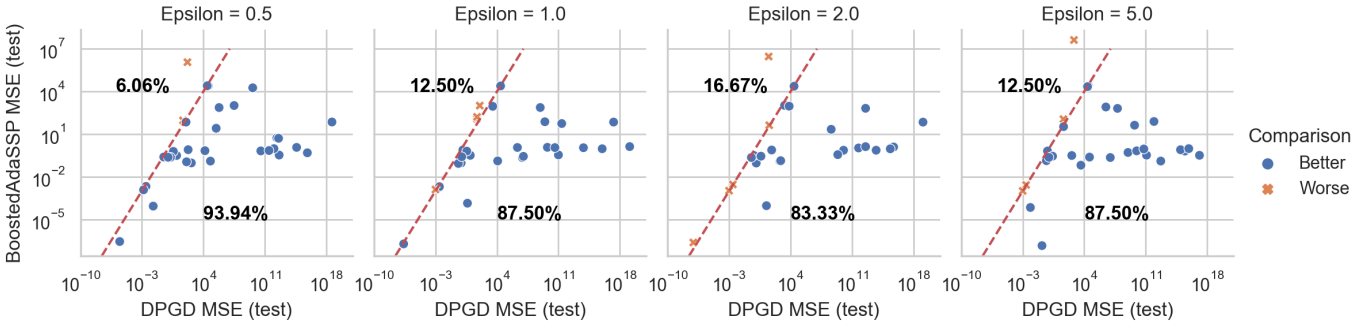
BoostedAdaSSP outperforms TukeyEM. BoostedAdaSSP also outperforms TukeyEM in all experimental regimes; we can see that the advantage that BoostedAdaSSP enjoys diminishes as the privacy parameter increases, since (when we optimize for the hyperparameters for both methods), both approach non-private (exact) linear regression. TukeyEM has only one hyperparameter, but it requires a massive number of data samples to train, due to its subsample-and-aggregate nature, and it produces an all-zero parameter vector in many scenarios. In contrast, our BoostedAdaSSP has only a couple more hyperparameters, and a common selection for them works well on many datasets. Comparisons are shown in Fig. 3.

With privacy, gradient boosting over linear models outperforms gradient boosting over tree based models. Results in Fig. 5 show that BoostedAdaSSP outperforms DB-EBM in all experimental regimes. DB-EBM is also a private gradient boosting algorithm, using tree based learners as base models. This is something that does not occur absent privacy (gradient boosting cannot improve on exact linear regression, as the update steps preserve linearity). This is emblematic of a more general message, that differential privacy rewards algorithmic simplicity (even when more complex algorithms outperform absent privacy constraints). This is because more complex algorithms require more noise addition for privacy, which is often ultimately not worth the tradeoff.

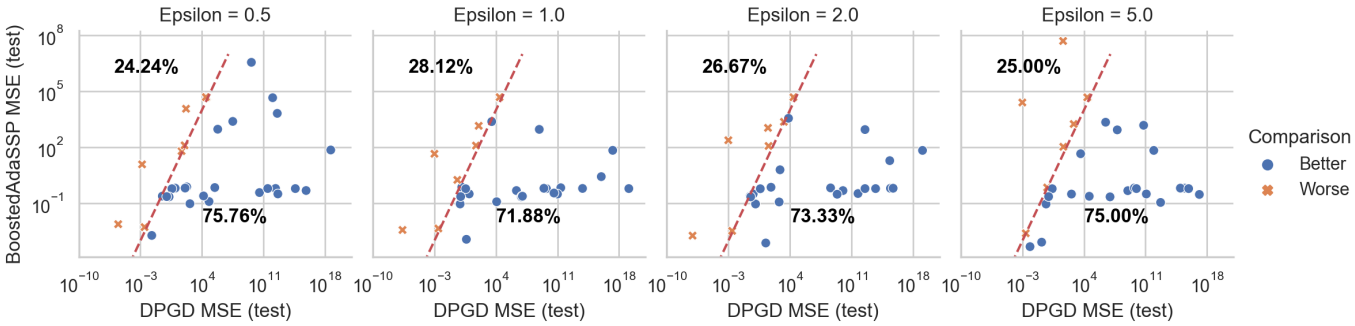
Theory work has been attempting at advancing differentially private linear regression with nearly optimal accuracy guarantees, and [6] notably provided two algorithms, but there is no public implementation of it. To demonstrate the empirical effectiveness of our work, we implemented the second algorithm in [6], namely DP-AMBSSGD (DP-Adaptive-Mini-Batch-Shuffled-SGD), with our best attempt, though the constants/hyperparameters in the algorithm are hard to realize. Therefore, for fair comparison, we only provide results with tuned hyperparameters for DP-AMBSSGD.

V. Robustness Results

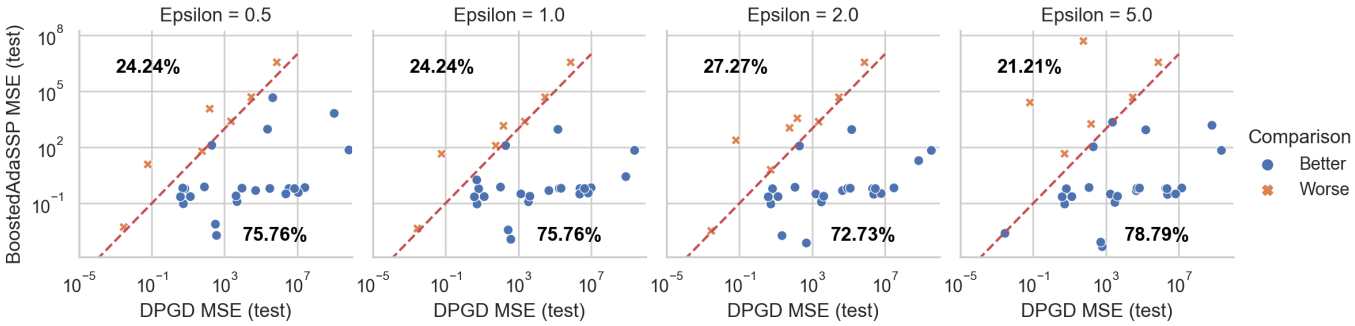
Since our algorithm directly leverages the idea of gradient boosting, it also benefits from the robustness of gradient boosting algorithm to outliers. This section presents prediction errors of our algorithm — BoostedAdaSSP — in the presence of outliers in the training set, in comparison to AdaSSP and TukeyEM.



(a) Non-privately Tuned BoostedAdaSSP vs. Non-privately Tuned DP Gradient Descent



(b) Fixed BoostedAdaSSP vs. Non-privately Tuned DP Gradient Descent



(c) Fixed BoostedAdaSSP vs. Fixed DP Gradient Descent

Fig. 2: **BoostedAdaSSP vs. DP Gradient Descent**. BoostedAdaSSP outperforms DP Gradient Descent in all comparisons, even when our algorithm uses a fixed set of hyperparameters.

The dataset is generated from the following linear regression model:

$$y = x^\top w + z, \tag{9}$$

where $w \in \mathbb{R}^d$, $x \sim N(0, I_{d\times d})$, $z \sim N(0, 0.1)$, and $I_{d\times d}$ is a $d \times d$ identify matrix. In our experiments, we fix d to be 10.

In our experiments, the total number of data samples vary from 10,000 to 1,000,000 to accommodate the desired setting for Tukey where the number of data samples should be 1,000 times larger than the dimension of samples. We also only consider high-privacy regime, where $\epsilon = 0.5$ and $\epsilon = 1$.

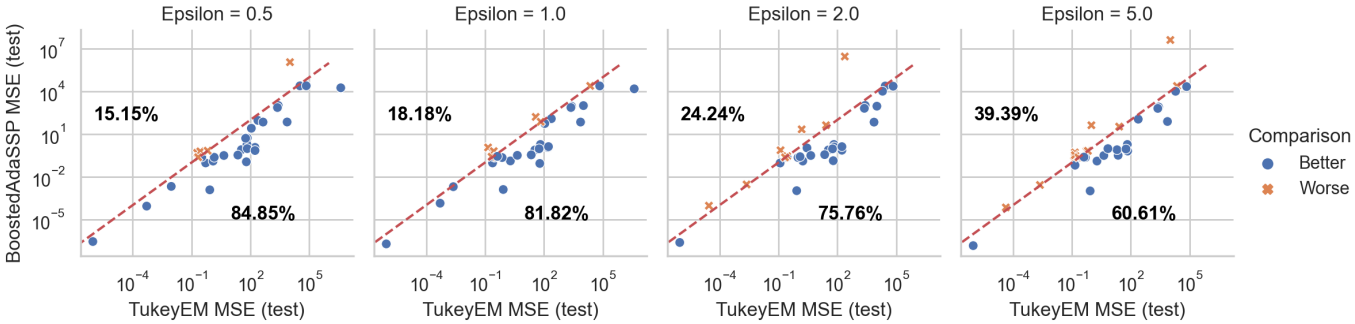
For data corruption, we consider the following scenarios, including introducing outliers to the label space, to the feature space, and outliers from the different linear models. We here present the data generation process for outliers and the performance of individual algorithms. All results are averaged over 10 random trials.

A. Outliers in the label space

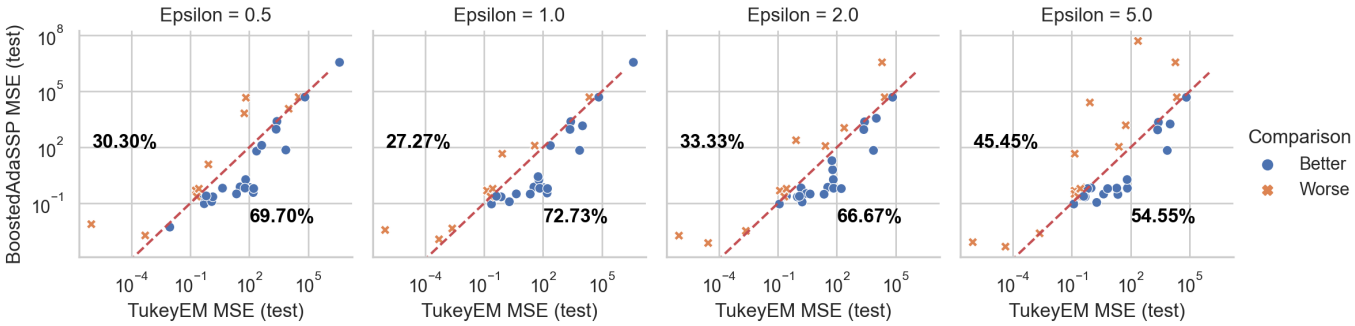
We first compare three different algorithms in the scenario where a small portion of labels are corrupted. Specifically, these labels are generated with much higher noise as follows:

$$y_{\text{out}} = x^\top w \times 10 + z, \tag{10}$$

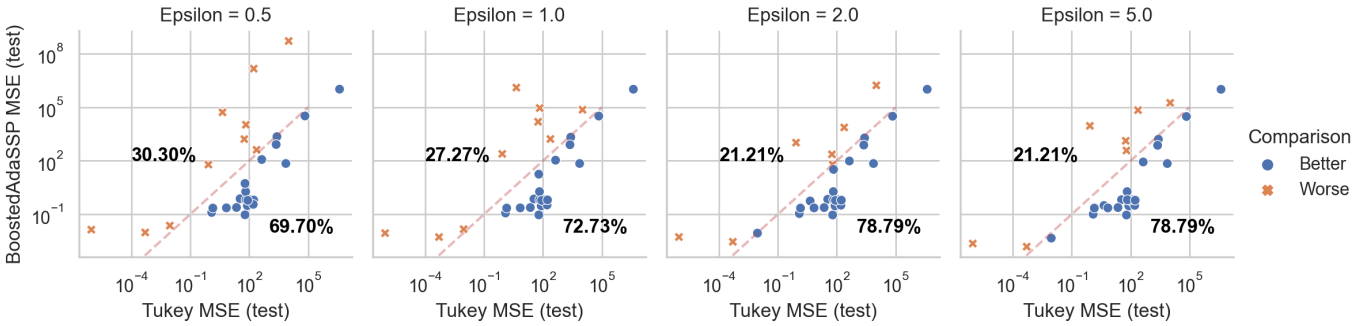
Fig. 6 and Fig. 7 present comparisons of our algorithm against AdaSSP and TukeyEM respectively. The first observation is that BoostedAdaSSP consistently outperforms AdaSSP when outliers are presented regardless of the private level, the total number of training samples, and the portion of outliers. Then, the second observation is that, when the portion of outliers is relatively low, i.e. 1%, and with a large number of



(a) Non-privately Tuned BoostedAdaSSP vs. Non-privately Tuned TukeyEM



(b) Fixed BoostedAdaSSP vs. Non-privately Tuned TukeyEM



(c) Fixed BoostedAdaSSP vs. Fixed TukeyEM

Fig. 3: **BoostedAdaSSP vs. TukeyEM.** BoostedAdaSSP outperforms TukeyEM in all comparisons, even when our algorithm uses a fixed set of hyperparameters. TukeyEM has the advantage of only having a single hyperparameter (number of models), however, in our experiments we find that there isn't a universally good selection for this hyperparameter.

training samples, TeukeyEM outperforms our BoostedAdaSSP algorithm. However, when the number of outliers increases, our algorithm consistently outperforms TukeyEM.

B. Outliers in the feature space

We then present a scenario where input features of a small portion of the dataset are corrupted, but the parameter vector stays the same. After the dataset is generated, the features of a small portion of samples are scaled up by 10, and it is shown as follows:

$$x_{\text{out}} = x \times 10, \tag{11}$$

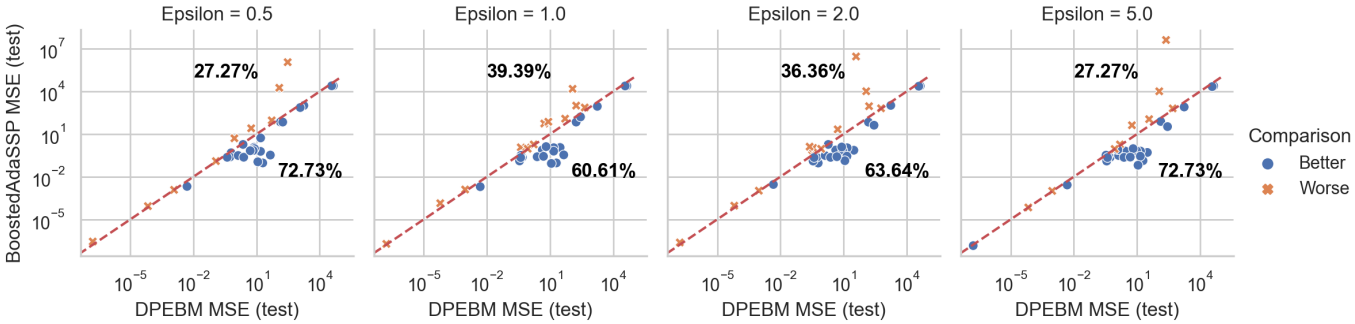
Fig. 8 presents the comparison of our algorithm and AdaSSP. The boosting nature of our algorithm improves the robustness to outliers, and the empirical evidence also reflects the improvement. Fig. 9 shows the comparison against TukeyEM, and the observation here is rather contradictory to that when the labels are corrupted. Here, when the number of samples are small and the corrupted samples are also small, TukeyEM outperforms, otherwise, ours dominates the performance.

C. Outliers from a different model

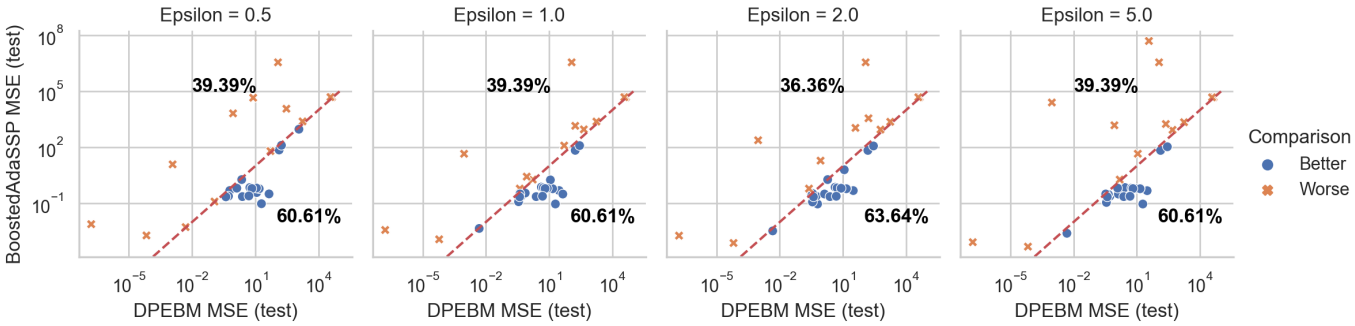
The last scenario that is worth considering is when a small portion of samples are produced from a different linear regression model. In our experiments, this other linear regression model is as follows:

$$y_{\text{out}} = x_{\text{out}}^{\top} w_{\text{out}} + z, \tag{12}$$

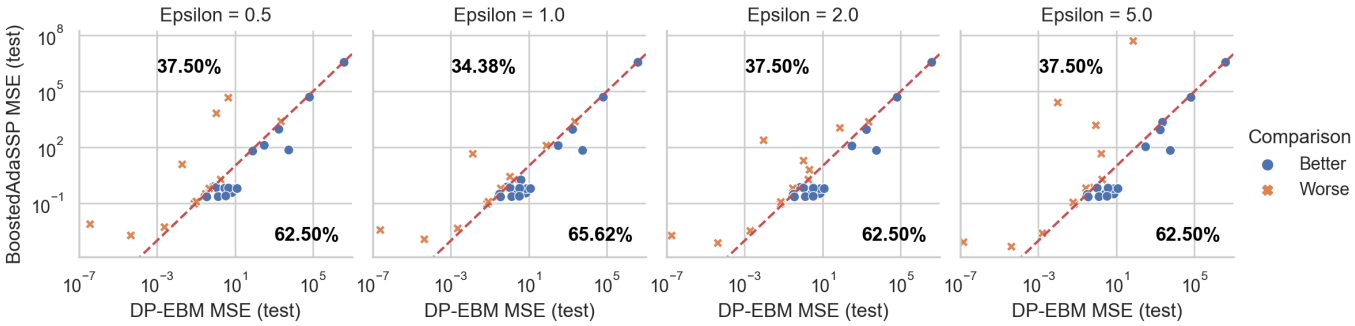
where $x_{\text{out}} \sim N(5, I_{d\times d})$, and $w_{\text{out}} = w + 5$. Fig. 10 and Fig. 11 shows the performance comparison of our BoostedAdaSSP



(a) Non-privately Tuned BoostedAdaSSP vs. Non-privately Tuned DP-EBM



(b) Fixed BoostedAdaSSP vs. Non-privately Tuned DP-EBM



(c) Fixed BoostedAdaSSP vs. Fixed DP-EBM

Fig. 4: **BoostedAdaSSP vs. DP-EBM.** BoostedAdaSSP and DP-EBM are both gradient boosting algorithms. BoostedAdaSSP uses linear models as the base class, whereas DP-EBM uses tree based models. Our method outperforms in all experimental regimes.

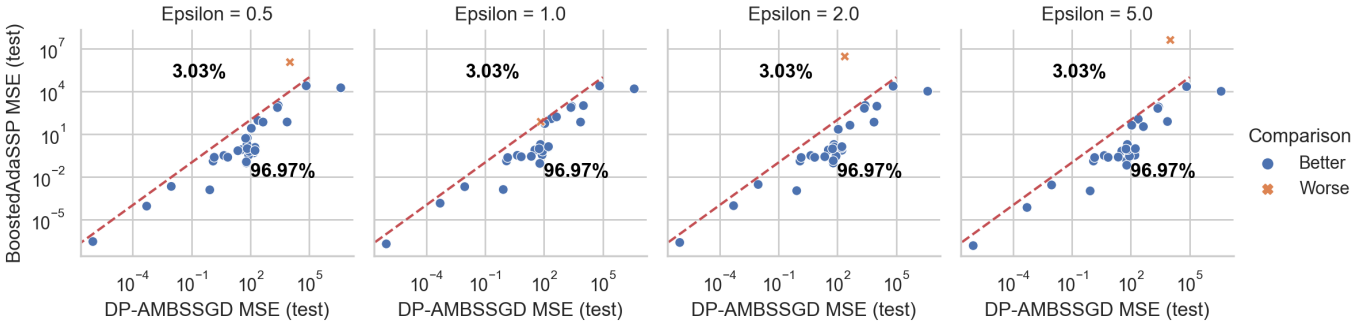
against AdaSSP and TukeyEM respectively. In this slightly more complicated scenario, boosting helps our algorithm overcome the impact of outliers, and outperforms AdaSSP. In addition, our algorithm is also more robust to outliers than TukeyEM.

VI. Theoretical Insights

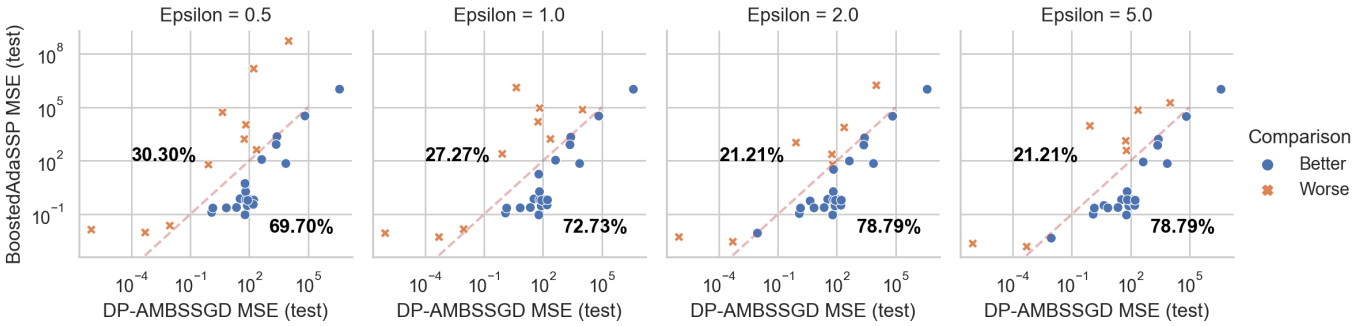
The improvement of BoostedAdaSSP over the base learner AdaSSP, from a theoretical perspective, can be attributed to the former's ability to adapt to arbitrary data clipping bounds. While the base learner AdaSSP is known to be optimal when the data clipping bounds are data-dependent and well-chosen ([4], Theorem 3), it suffers from significant bias when the data clipping bounds are mis-specified (i.e. much closer to 0 relative to the data range).

This bias exists even in the simplest "zero-dimensional" case where linear regression reduces to estimating the population mean of real-valued data. Consider a data set $Y_1, Y_2, \cdots, Y_n \overset{i.i.d.}{\sim} \mathcal{N}(\mu, 1)$. With $C_\tau(a) = a \min(1, |a|/\tau)$ denoting the clipping operator, the zero-dimensional AdaSSP estimator is simply $\hat{\mu}_1 = n^{-1} \sum_{i \in [n]} C_\tau(Y_i) + Z$, where Z is the requisite Gaussian noise for differential privacy. The bias of the AdaSSP estimator, $|\mathbb{E}\hat{\mu}_1 - \mu|$, is then at least $|\mu| - \tau$, since $|\mathbb{E}\hat{\mu}_1| \leq \tau$.

In contrast, the BoostedAdaSSP algorithm converges to the population mean μ for any non-zero clipping bound τ. The theoretical analysis is in Section E of the appendix.



(a) Non-privately Tuned BoostedAdaSSP vs. Non-privately Tuned DP-AMBSSGD



(b) Fixed BoostedAdaSSP vs. Non-privately Tuned DP-AMBSSGD

Fig. 5: **BoostedAdaSSP vs. DP-AMBSSGD.** BoostedAdaSSP and DP-AMBSSGD are both iterative algorithms. DP-AMBSSGD adaptively chooses the clipping bound in each iterative to achieve theoretically optimal performance on linear regression. We, with our best attempt, implemented the algorithm, and only presented results with tuned hyperparameters using Optuna in a non-private way. As we can see, our algorithm BoostedAdaSSP with optimally tuned hyperparameters outperforms optimally tuned DP-AMBSSGD, and even with fixed hyperparameters, our algorithm still achieves lower MSE values on a majority of datasets.

VII. Conclusion

We present an augmentation of AdaSSP via gradient boosting for differentially private linear regression, and we name it BoostedAdaSSP. Through extensive experiments and comparisons against other approaches, we show that BoostedAdaSSP is less sensitive to hyperparameter choices than other approaches, and with fixed hyperparameters, BoostedAdaSSP outperforms other algorithms on a majority of real datasets. Moreover, on synthetic data, we demonstrate the robustness of our algorithm, BoostedAdaSSP, to outliers in the training set. Overall, our algorithm is a practical and robust solution to situations where hyperparameters of linear regression problems are hard to determine.

References

[1] X. Wu, M. Fredrikson, W. Wu, S. Jha, and J. F. Naughton, "Revisiting differentially private regression: Lessons from learning theory and their consequences," *arXiv preprint arXiv:1512.06388*, 2015.

[2] O. Sheffet, "Differentially private ordinary least squares," in *International Conference on Machine Learning*, pp. 3105–3114, PMLR, 2017.

[3] O. Sheffet, "Old techniques in differentially private linear regression," in *Algorithmic Learning Theory*, pp. 789–827, PMLR, 2019.

[4] Y.-X. Wang, "Revisiting differentially private linear regression: optimal and adaptive prediction & estimation in unbounded domain," *Uncertainty in Artificial Intelligence (UAI-18)*, 2018.

[5] D. Alabi, A. McMillan, J. Sarathy, A. Smith, and S. Vadhan, "Differentially private simple linear regression," *Proceedings on Privacy Enhancing Technologies*, vol. 2, pp. 184–204, 2022.

[6] P. Varshney, A. Thakurta, and P. Jain, "(nearly) optimal private linear regression for sub-gaussian data via adaptive clipping," in *Conference on Learning Theory*, pp. 1126–1166, PMLR, 2022.

[7] K. Amin, M. Joseph, M. Ribero, and S. Vassilvitskii, "Easy differentially private linear regression," *arXiv preprint arXiv:2208.07353*, 2022.

[8] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[9] H. Nori, R. Caruana, Z. Bu, J. H. Shen, and J. Kulkarni, "Accuracy, interpretability, and differential privacy via explainable boosting," in *International Conference on Machine Learning*, pp. 8227–8237, PMLR, 2021.

[10] D. Kifer, A. Smith, and A. Thakurta, "Private convex empirical risk minimization and high-dimensional regression," in *Conference on Learning Theory*, pp. 25–1, JMLR Workshop and Conference Proceedings, 2012.

[11] R. Bassily, A. Smith, and A. Thakurta, "Private empirical risk minimization: Efficient algorithms and tight error bounds," in *2014 IEEE 55th annual symposium on foundations of computer science*, pp. 464–473, IEEE, 2014.

[12] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

[13] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization.," *Journal of Machine Learning Research*, vol. 12, no. 3, 2011.

[14] D. Vu and A. Slavkovic, "Differential privacy for clinical trial data: Preliminary evaluations," in *2009 IEEE International Conference on*

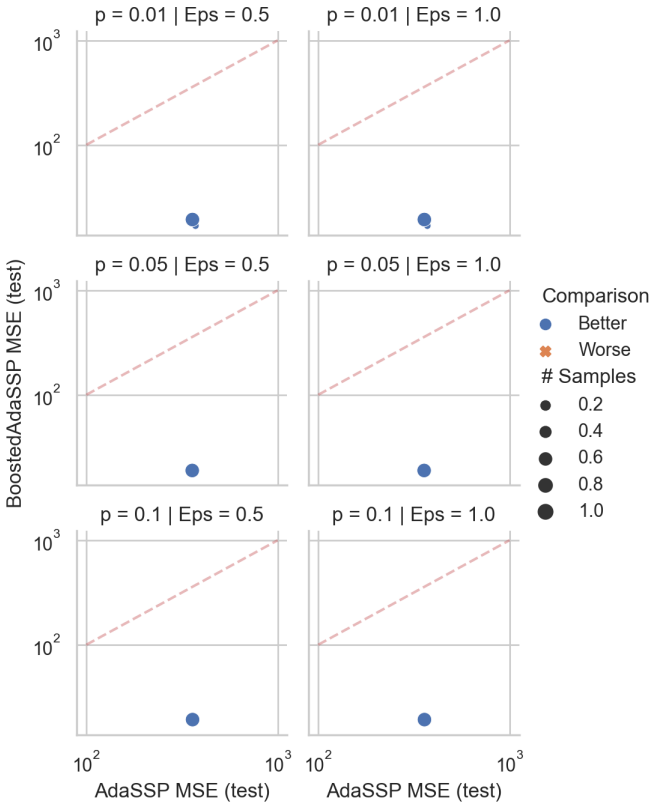


Fig. 6: Label Corruption, BoostedAdaSSP vs. AdaSSP. It is easy to see that BoostedAdaSSP outperforms AdaSSP when a small portion of the labels are corrupted. p refers to the portion of corrupted labels, and the unit of '# Samples' is a million.

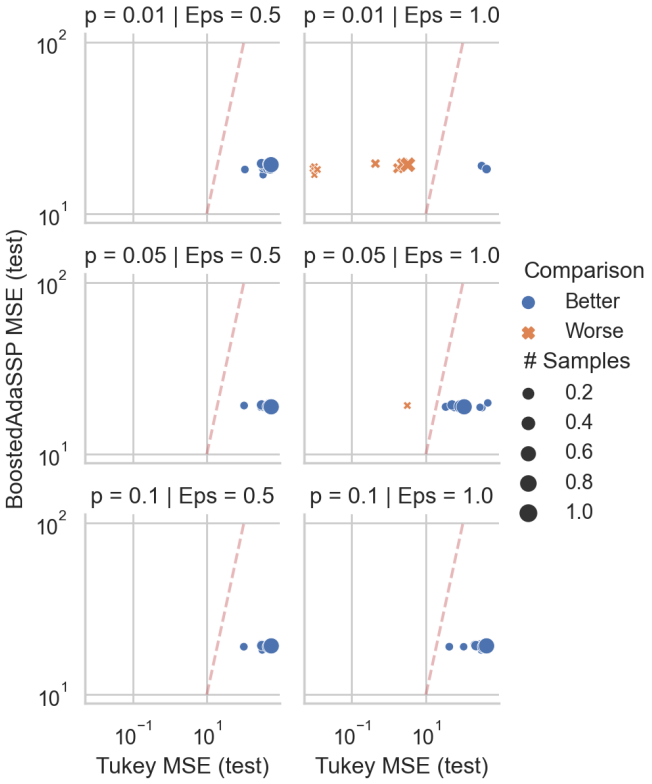


Fig. 7: Label Corruption, BoostedAdaSSP vs. TukeyEM. When the corruption is only limited to a tiny portion of the data, and the number of samples is large, TukeyEM outperforms ours. While in all other scenarios, ours is more robusted to outliers.

Data Mining Workshops, pp. 138–143, IEEE, 2009.

[15] Q. Li, Z. Wu, Z. Wen, and B. He, "Privacy-preserving gradient boosting decision trees," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 784–791, 2020.

[16] N. Grislain and J. Gonzalvez, "Dp-xgboost: Private machine learning at scale," *arXiv preprint arXiv:2110.12770*, 2021.

[17] J. Liu and K. Talwar, "Private selection from private candidates," in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pp. 298–309, 2019.

[18] S. Mohapatra, S. Sasy, X. He, G. Kamath, and O. Thakkar, "The role of adaptive optimizers for honest private hyperparameter selection," in *Proceedings of the aaai conference on artificial intelligence*, vol. 36, pp. 7806–7813, 2022.

[19] A. Koskela and T. Kulkarni, "Practical differentially private hyperparameter tuning with subsampling," *arXiv preprint arXiv:2301.11989*, 2023.

[20] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proceedings of the 3rd Conference on Theory of Cryptography*, TCC '06, pp. 265–284, 2006.

[21] J. Dong, A. Roth, and W. Su, "Gaussian differential privacy," *Journal of the Royal Statistical Society*, 2021.

[22] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on tabular data?," *arXiv preprint arXiv:2207.08815*, 2022.

[23] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[24] T. T. Cai, Y. Wang, and L. Zhang, "The cost of privacy: Optimal rates of convergence for parameter estimation with differential privacy," *Annals of Statistics*, vol. 49, no. 5, pp. 2825–2850, 2021.

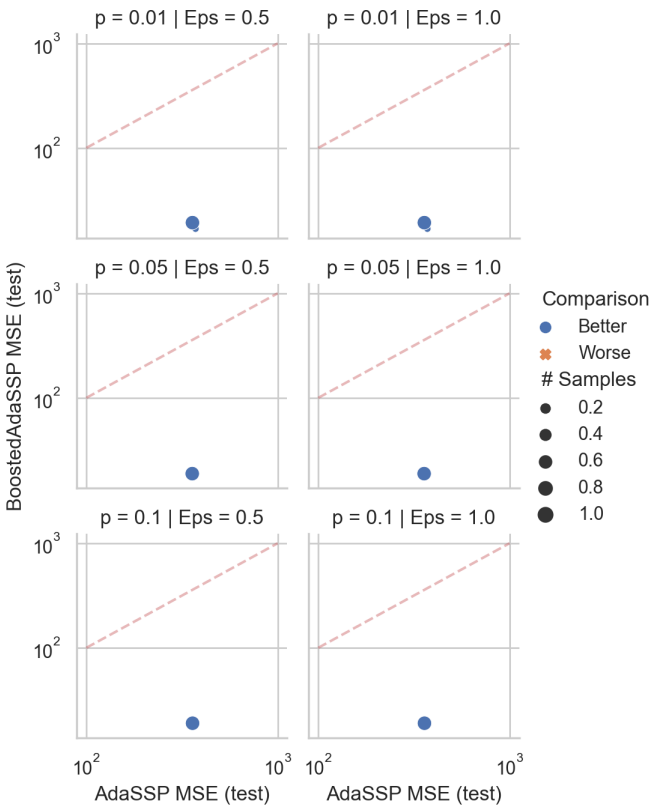


Fig. 8: Feature Corruption, BoostedAdaSSP vs. AdaSSP. The iterative approach of our algorithm improves the performance of AdaSSP even in the case where features of a small portion of data samples are corrupted.

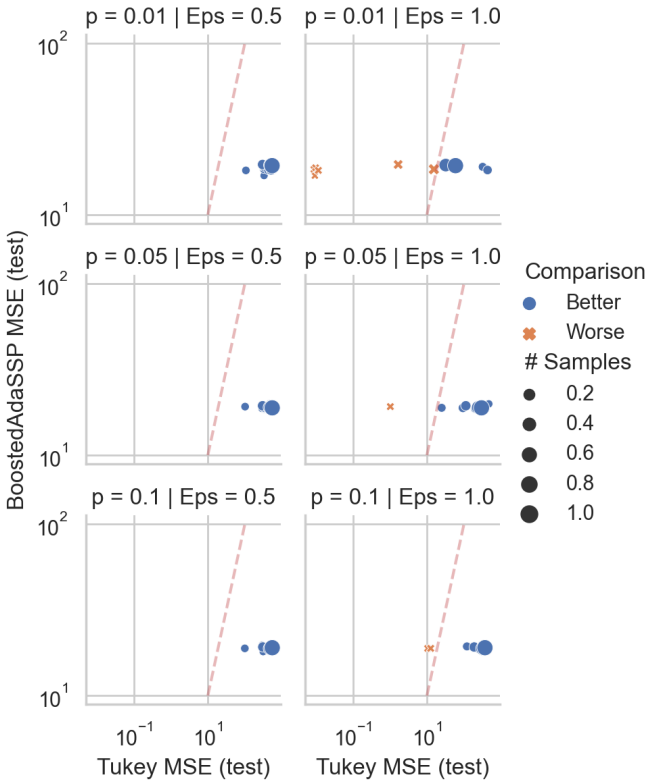


Fig. 9: Feature Corruption, BoostedAdaSSP vs. TukeyEM. When higher privacy budget is allocated, TukeyEM becomes more resilient to outliers in the training set when the total number of samples is small. However, ours still outperforms TukeyEM in other situations.

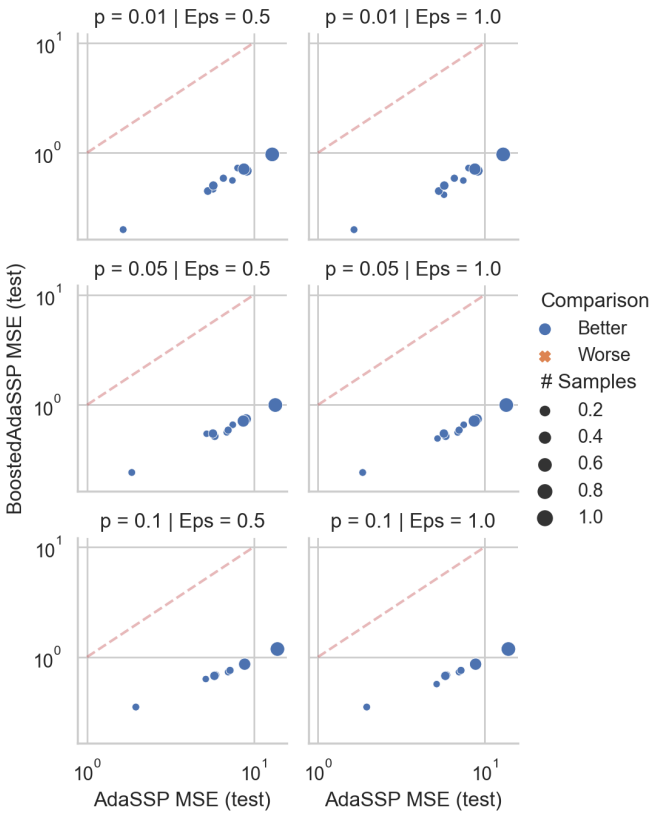


Fig. 10: Model Corruption, BoostedAdaSSP vs. AdaSSP.

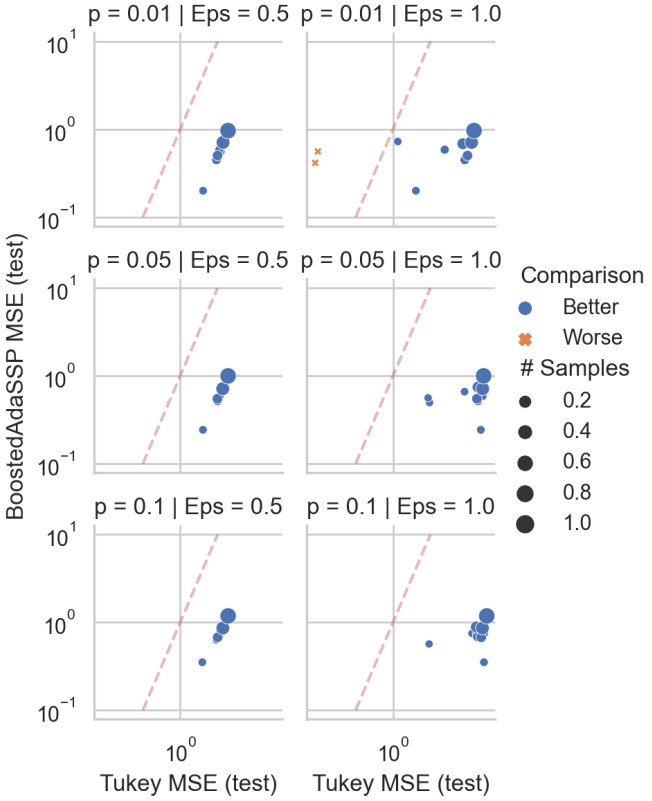


Fig. 11: Model Corruption, BoostedAdaSSP vs. TukeyEM. When a small fraction of data samples are generated from a different linear model, our algorithm BoostedAdaSSP is more robust to these type of outliers than Tukey.

APPENDIX

A. Hyperparameters

TABLE I: Hyperparameters of individual algorithms, and their individual tuning ranges.

Hyperparameters	Data Clipping Bound	Gradient Clipping Bound	# Models /Iterations /Updates	Learning Rate Schedule	# Leaves
Tuning Range	[1e-5,1e+5]	[1e-5,1e+5]	[1,4000]	$\{1, t^{-1}, t^{-1/2}\}$	[1,1000]
Algorithm					
BoostedAdaSSP	Yes	No	Yes	Yes	No
AdaSSP	Yes	No	No	No	No
DP-Gradient Descent	No	Yes	Yes	Yes	No
TukeyEM	No	No	Yes	No	No
DP-EBM	Yes	No	Yes	No	Yes
Fixed Hyperparameters					
BoostedAdaSSP	1	-	100	1	-

B. AdaSSP Algorithm for Learning a Single Ridge Regressor

Let $\widehat{\cdot}$ denote private versions of the corresponding statistics. Then, AdaSSP privately releases the sufficient statistics of ridge regressor as follows.

Algorithm 2 Private Ridge regression via AdaSSP(data X, y, calibration ratio a, b, c , Privacy parameter ϵ, δ, Bound on $||\mathcal{X}||, ||\mathcal{Y}||$)

Find μ such that μ-GDP satisfies (ϵ, δ)-DP. `# Corollary II.5`
Calibrate μ_1, μ_2, μ_3 such that $\mu_1 : \mu_2 : \mu_3 = a : b : c$ and $\mu = \sqrt{\mu_1^2 + \mu_2^2 + \mu_3^2}$.
Clip X so that $\max_i ||x_i|| \leq ||\mathcal{X}||$
Clip y so that $\max_i ||y_i|| \leq ||\mathcal{Y}||$
$\widehat{X^\top X} = GM(X^\top X, \mu_1, ||\mathcal{X}||)$
$\widehat{X^\top y} = GM(X^\top y, \mu_2, \sqrt{||\mathcal{X}||||\mathcal{Y}||})$
$\widehat{\lambda} = GM(\lambda_{\min}(X^\top X), \mu_3, ||\mathcal{X}||)$
Output $\widehat{\theta}_t = (\widehat{X^\top X} + \widehat{\lambda} I)^{-1} \widehat{X^\top y}$

Algorithm 2 instantiates three Gaussian mechanisms with μ_1, μ_2, and μ_3 to privately release each sufficient statistic. Hence the composition

$$\widehat{\theta}_t = (\widehat{X^\top X} + \widehat{\lambda} I)^{-1} \widehat{X^\top g_t} \tag{13}$$

is (ϵ, δ)-DP. Detailed proof is available in Theorem 3 of [4].

C. Datasets

All 33 datasets in our experiments come from OpenML. Task information is listed in Tab. II.

TABLE II: **Regression Tasks.** 'Task ID' refers to the task id on OpenML, n refers to the number of total data points, d refers to the dimension of features after one-hot encoding the categorical features, and 'True Bound' indicates the maximum absolute value of labels.

Task ID	n	# Columns	d	True Bound	n / d
361072	8192	21	21	99.000000	390.095238
361073	15000	26	26	100.000000	576.923077
361074	16599	16	16	0.078000	1037.437500
361075	7797	613	613	26.000000	12.719413
361076	6497	11	11	9.000000	590.636364
361077	13750	33	33	0.003600	416.666667
361078	20640	8	8	13.122367	2580.000000
361079	22784	16	16	13.122367	1424.000000
361080	53940	6	6	9.842888	8990.000000
361081	10692	8	8	13.928840	1336.500000
361082	17379	6	6	977.000000	2896.500000
361083	581835	9	9	5.528238	64648.333333
361084	21613	15	15	15.856731	1440.866667
361085	10081	6	6	1.000000	1680.166667
361086	163065	3	3	11.958631	54355.000000
361087	13932	13	13	14.790071	1071.692308
361088	21263	79	79	185.000000	269.151899
361089	20640	8	8	1.791761	2580.000000
361090	18063	5	5	12.765691	3612.600000
361091	515345	90	90	2011.000000	5726.055556
361092	8885	62	82	1.000000	108.353659
361093	4052	7	12	2.300000	337.666667
361094	8641	4	5	40.000000	1728.200000
361095	166821	9	23	10.084141	7253.086957
361096	53940	9	26	9.842888	2074.615385
361097	4209	359	735	265.320000	5.726531
361098	10692	11	17	13.928840	628.941176
361099	17379	11	20	977.000000	868.950000
361100	39644	59	73	13.645079	543.068493
361101	581835	16	31	5.528238	18768.870968
361102	21613	17	19	15.856731	1137.526316
361103	394299	6	26	6.480505	15165.346154
361104	241600	9	15	8.113915	16106.666667

D. Impact of the Ratio N/d

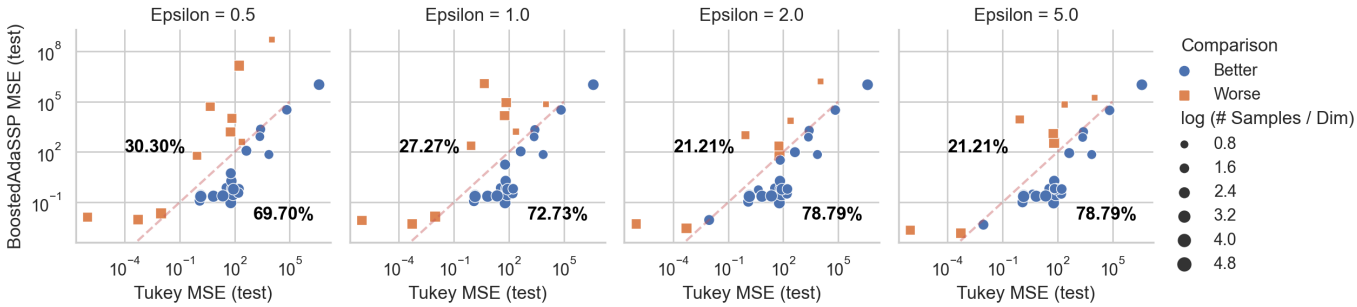


Fig. 12: The plot shows the comparison between our algorithm, BoostedAdaSSP, versus TukeyEM when both algorithms use fixed and shared hyperparameters across datasets. Specifically, the size of each marker represents the ratio between the number of samples of a dataset and the dimension of features. It doesn't seem obvious that the ratio N/d is a controlling factor for the success of an algorithm.

E. Omitted Material in Section VI

The zero-dimensional BoostedAdaSSP algorithm for estimating μ from $Y_1, Y_2, \cdots, Y_n \overset{i.i.d.}{\sim} \mathcal{N}(\mu, 1)$ is defined in Algorithm 3.

Theorem A.1. *For every $\tau = O(1)$ not depending on sample size n, Algorithm 3 is Gaussian DP with parameter ρ, and there exists a data-independent choice of number of boosting rounds R such that the estimator $\hat{\mu}_R$ converges to the true parameter μ, with the rate of convergence*

$$\mathbb{E}|\hat{\mu}_R - \mu| = O\left(\frac{\log n}{\sqrt{n}} + \frac{\log^{3/2} n}{n\sqrt{\rho}}\right). \tag{14}$$

Proof of Theorem A.1. The Gaussian DP of Algorithm 3 follows from the Gaussian mechanism II.3 and the composition theorem II.4, by observing that the sensitivity of the clipped sample mean is $2\tau/n$. Next, we establish the convergence of $\hat{\mu}_R$ by comparing with Algorithm 4, an "infinite-sample" version of Algorithm 3. □

Algorithm 3 Zero-dimensional BoostedAdaSSP

Require: Clipping function C_τ, number of rounds R, Gaussian DP parameter ρ.
Require: Data $Y_1, Y_2, \cdots, Y_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu, 1)$.
Initialize: $\hat{\mu}_0 = 0$
for $t \in [R]$ **do**
Compute DP residual mean

$$\hat{\mu}_j = \hat{\mu}_{j-1} + \frac{1}{n}\sum_{i\in[n]} C_\tau(Y_i - \hat{\mu}_{j-1}) + Z_j, \tag{15}$$

where $Z_j \sim \mathcal{N}\left(0, \frac{4R\tau^2}{n^2\rho}\right)$.
end for
Output $\hat{\mu}_R$

Algorithm 4 Infinite-sample algorithm

Require: Clipping function C_τ, number of rounds R.
Require: Infinite samples from $\mathcal{N}(\mu, 1)$.
Initialize: $\theta_0 = 0$
for $t \in [R]$ **do**
Compute true truncated residual mean

$$\theta_j = \theta_{j-1} + \mathbb{E}_{Y\sim\mathcal{N}(\mu,1)} C_\tau(Y - \theta_{j-1}). \tag{16}$$

end for
Output θ_R

By considering an idealized "infinite sample" setting where we have access to true distributional quantities $\{\mathbb{E}C_\tau(Y - \theta_{j-1})\}_{j\in R}$, Algorithm 4 removes all the randomness in the finite-sample Algorithm 3 and allows us to focus entirely on the bias-reduction effect of boosting. Indeed, the infinite-sample "estimator" θ_R converges deterministically to μ.

Proposition A.2. *Suppose the number of rounds $R > \frac{\max(0,|\mu|-\tau)}{(\Phi(2\tau)-1/2)\tau}$. The error of θ_R is bounded by*

$$|\theta_R - \mu| \le \tau\left(3/2 - \Phi(\tau)\right)^{R - \frac{\max(0,|\mu|-\tau)}{(\Phi(2\tau)-1/2)\tau}}. \tag{17}$$

That is, after a warm-up of $\frac{\max(0,|\mu|-\tau)}{(\Phi(2\tau)-1/2)\tau}$ rounds, the error of θ_R decays geometrically fast, as $0 < 3/2 - \Phi(\tau) < 1$ for any $\tau > 0$. It now suffices to bound the difference $|\theta_R - \hat{\mu}_R|$.

Proposition A.3. *The difference between outputs of Algorithms 3 and 4 is bounded by*

$$\mathbb{E}|\hat{\mu}_R - \theta_R| = O\left(\frac{R\tau}{\sqrt{n}} + \frac{R^{3/2}\tau}{n\sqrt{\rho}}\right). \tag{18}$$

By choosing an $R = O(\log n)$ and $R > \frac{\max(0,|\mu|-\tau)}{(\Phi(2\tau)-1/2)\tau}$, we have $|\theta_R - \mu| = O(\tau/n)$ by Proposition A.2, and then

$$\mathbb{E}|\hat{\mu}_R - \mu| \le |\theta_R - \mu| + \mathbb{E}|\hat{\mu}_R - \theta_R| \tag{19}$$

$$= O\left(\frac{\tau}{n}\right) + O\left(\frac{\tau \log n}{\sqrt{n}} + \frac{\tau \log^{3/2} n}{n\sqrt{\rho}}\right). \tag{20}$$

As $\tau = O(1)$ by assumption, the main proof is complete. Propositions A.2 and A.3 are proved in Section E.

1) Proof of Proposition A.2: Let $\Delta_t = \theta_{t-1} - \mu$, so that $|\Delta_t|$ is the bias of Algorithm 4 after $t-1$ iterations. We would like to see $|\Delta_t|$ decay as quickly as possible in t. The following lemmas quantify the rate of decay.

Lemma A.4. *Let $t \ge 0$ and τ be the clipping threshold. Suppose $|\Delta_t| \le \tau$. Then $|\Delta_{t+1}| \le \left(\frac{3}{2} - \Phi(\tau)\right)|\Delta_t|$, where $\Phi(\cdot)$ is the standard Gaussian CDF.*

Lemma A.5. *Let $t \ge 0$ and τ be the clipping threshold. Suppose $|\Delta_t| > \tau$. Then $|\Delta_{t+1}| \le |\Delta_t| - (\Phi(2\tau) - 1/2)\tau$, where $\Phi(\cdot)$ is the standard Gaussian CDF.*

The lemmas taken together suggest that, if $|\mu| > \tau$, then it takes $\frac{|\mu|-\tau}{(\Phi(2\tau)-1/2)\tau}$ rounds for the error $|\Delta_t|$ to decrease below τ. As soon as $|\Delta_t| \le \tau$, then $|\Delta_t|$ decays geometrically by a factor of $(3/2 - \Phi(\tau))$ each round. Then, for $R > \frac{\max(0,|\mu|-\tau)}{(\Phi(2\tau)-1/2)\tau}$, the desired bound in Proposition A.2 follows.

It remains to prove the two lemmas.

Proof of Lemma A.4. Let m_t denote the true truncated residual mean in the t-th step.

Throughout the proof, let P_t denote $\mathcal{N}(\Delta_t, 1)$. Note that if $\mu_t = 0$, then $m_t = \Delta_t = \Delta_{t+1}$, and so the inequality holds.

Now we consider the case where $\Delta_t \ne 0$. Without loss of generality, assume $\mu_t > 0$. Let C denote the clipping operation, that is $C(Y) = Y \min(1, \tau/|Y|)$. Then we can decompose the estimate m_t as follows:

$$m_t = \mathbb{E}_{P_t}[C(Y)] \tag{21}$$

$$= \underbrace{\Pr_{P_t}[Y < -\tau](-\tau) + \Pr_{P_t}[Y > \tau + 2\Delta_t]\tau}_{T_1}$$

$$+ \underbrace{\Pr_{P_t}[Y \in [-\tau, \tau + 2\Delta_t]]\mathbb{E}[C(Y) \mid Y \in [-\tau, \tau + 2\Delta_t]]}_{T_2} \tag{22}$$

Since the distribution P_t is symmetric about Δ_t, $T_1 = 0$. Now we further decompose T_2 by considering three different intervals:

$$T_2 = \underbrace{\Pr_{P_t}[Y \in [2\Delta_t - \tau, \tau]]\mathbb{E}_{P_t}[C(Y) \mid Y \in [2\Delta_t - \tau, \tau]]}_{T_3}$$

$$+ \underbrace{\Pr_{P_t}[Y \in [-\tau, 2\Delta_t - \tau]]\mathbb{E}_{P_t}[C(Y) \mid Y \in [-\tau, 2\Delta_t - \tau]]}_{T_4}$$

$$+ \underbrace{\Pr_{P_t}[Y \in [\tau, \tau + 2\Delta_t]]\mathbb{E}_{P_t}[C(Y) \mid Y \in [\tau, \tau + 2\Delta_t]]}_{T_5} \tag{23}$$

In the interval of T_3, no Y is clipped, so $C(Y) = Y$. Since the interval is also centered at the mean Δ_t, the conditional expectation is Δ_t, so

$$T_3 = \Pr_{P_t}[Y \in [2\Delta_t - \tau, \tau]]\Delta_t. \tag{24}$$

In the interval of T_5, each Y is clipped, so

$$T_5 = \Pr_{P_t}[Y \in [\tau, \tau + 2\Delta_t]]\tau. \tag{25}$$

In the interval of T_4, no Y is clipped. Moreover, for any $y, y' \in [-\tau, 2\Delta_t - \tau]$ such that $y > y'$, the density $P_t(y) > P_t(y')$. This allows us to lower bound T_4:

$$\begin{aligned}
T_4 &= \int_{[-\tau, 2\Delta_t - \tau]} Y\, P_t(Y) \\
&= \int_{[\Delta_t - \tau, 2\Delta_t - \tau]} Y\, P_t(Y) + \int_{[-\tau, \Delta_t - \tau]} Y\, P_t(Y) \\
&= \int_{[\Delta_t - \tau, 2\Delta_t - \tau]} [Y\, P_t(Y) + (2\Delta_t - 2\tau - Y) P_t(2\Delta_t - 2\tau - Y)] \\
&\geq \int_{[\Delta_t - \tau, 2\Delta_t - \tau]} (\Delta_t - \tau)\, (P_t(Y) + P_t(2\Delta_t - 2\tau - Y)) && (26) \\
&= (\Delta_t - \tau) \int_{[-\tau, 2\Delta_t - \tau]} P_t(Y) \\
&= (\Delta_t - \tau) \Pr_{P_t}[Y \in [-\tau, 2\Delta_t - \tau]] && (27)
\end{aligned}$$

where the step in inequality (26) follows from the fact that for any four numbers a, b, c, d such that $a \geq c$ and $b \geq d$, then $ab + cd \geq \frac{(a+c)}{2}(b + d)$. Finally, note that $\Pr_{P_t}[Y \in [-\tau, 2\Delta_t - \tau]] = \Pr_{P_t}[Y \in [\tau, 2\Delta_t + \tau]]$ since the two intervals are symmetric about Δ_t. Thus,

$$T_4 + T_5 = \frac{\Delta_t}{2} \left(\Pr_{P_t}[Y \in [-\tau, 2\Delta_t - \tau]] + \Pr_{P_t}[Y \in [\tau, 2\Delta_t + \tau]] \right) \tag{28}$$

Putting (24), (25) and (27) together, we get

$$m_t = T_3 + T_4 + T_5 \geq \frac{\Delta_t}{2} \Pr_{P_t}[Y \in [-\tau, \tau + 2\Delta_t]] \tag{29}$$

Finally, note that

$$\begin{aligned}
\Pr_{P_t}[Y \in [-\tau, \tau + 2\Delta_t]] &= \Pr_{P_t}[Y - \Delta_t \in [-\tau - \Delta_t, \tau + \Delta_t]] && (30) \\
&\geq \Pr_{P_t}[Y - \Delta_t \in [-\tau, \tau]] && (31) \\
&= 2\Phi(\tau) - 1 && (32)
\end{aligned}$$

This means

$$\Delta_{t+1} = \Delta_t - m_t \leq (3/2 - \Phi(\tau))\Delta_t \tag{33}$$

which completes the proof. □

Proof of Lemma A.5. Throughout the proof, let P_t denote $\mathcal{N}(\Delta_t, 1)$. Without the loss of generality, assume $\Delta_t > 0$. We start by decomposing the mean of the clipped distribution:

$$\begin{aligned}
m_t &= \mathbb{E}_{P_t}[C(Y)] && (34) \\
&= \underbrace{\Pr_{P_t}[Y < -\tau](-\tau) + \Pr_{P_t}[Y > \tau + 2\Delta_t]\tau}_{T_1} \\
&\quad + \underbrace{\Pr_{P_t}[Y \in [-\tau, \tau + 2\Delta_t]] \mathbb{E}[C(Y) \mid Y \in [-\tau, \tau + 2\Delta_t]]}_{T_2} && (35)
\end{aligned}$$

By the symmetric property of P_t, $T_1 = 0$. Now we will further break down T_2 into the parts that were unclipped and clipped:

$$\begin{aligned}
T_2 &= \underbrace{\Pr_{P_t}[Y \in [-\tau, \tau]] \mathbb{E}[Y \mid Y \in [-\tau, \tau]]}_{T_3} \\
&\quad + \underbrace{\Pr_{P_t}[Y \in [\tau, 2\Delta_t + \tau]]\tau}_{T_4} && (36)
\end{aligned}$$

First, note that for each $y \in [0, \tau]$, we have $P_t(y) \geq P_t(-y)$ since y is closer than $-y$ to the mean Δ_t. This implies

$$\begin{aligned} T_3 &= \int_{[-\tau,\tau]} Y\, P_t(Y) \\ &= \int_{[0,\tau]} [Y\, P_t(Y) + (-Y)\, P_t(-Y)] \\ &\geq \int_{[0,\tau]} [Y\, P_t(Y) + (-Y)\, P_t(Y)] = 0 \end{aligned} \tag{37}$$

Finally, we note that in T_4, the probability of interval can be lower bounded as:

$$\begin{aligned} \Pr_{P_t}[Y \in [\tau, 2\Delta_t + \tau]] &= \Pr_{Z \sim \mathcal{N}(0,1)}[Z \in [\tau - \Delta_t, \Delta_t + \tau]] \\ &\geq \Pr_{Z \sim \mathcal{N}(0,1)}[Z \in [0, 2\tau]] \end{aligned} \tag{38}$$

where the last inequality follows from $\tau \leq \Delta_t$. Thus, $m_t \geq (\Phi(2\tau) - 1/2)\tau$, which recovers the stated bound. □

2) Proof of Proposition A.3: Define

$$A_{j,n} := \frac{1}{n} \sum_{i \in [n]} C_\tau(Y_i - \theta_j) - \mathbb{E} C_\tau(Y - \theta_j), \quad B_{j,n} := \frac{1}{n} \sum_{i \in [n]} \left(C_\tau(Y_i - \hat{\mu}_j) - C_\tau(Y_i - \theta_j) \right), \tag{39}$$

then for every $j \in [R]$ it holds, by equations (15) and (16), that

$$\hat{\mu}_j - \theta_j = \hat{\mu}_{j-1} - \theta_{j-1} + A_{j-1,n} + B_{j-1,n} + Z_j. \tag{40}$$

To simplify the right side, observe that:

- each term in $B_{j-1,n}$, $C_\tau(Y_i - \hat{\mu}_{j-1}) - C_\tau(Y_i - \theta_{j-1})$ is of the same sign as $(Y_i - \hat{\mu}_{j-1}) - (Y_i - \theta_{j-1}) = \theta_{j-1} - \hat{\mu}_{j-1}$, since clipping preserves ordering: if $a \leq b$, then $C_\tau(a) \leq C_\tau(b)$;
- the magnitude $|C_\tau(Y_i - \hat{\mu}_{j-1}) - C_\tau(Y_i - \theta_{j-1})|$ is upper bounded by $|(Y_i - \hat{\mu}_{j-1}) - (Y_i - \theta_{j-1})| = |\theta_{j-1} - \hat{\mu}_{j-1}|$, as clipping is non-expansive: for any a, b, $|C_\tau(a) - C_\tau(b)| \leq |a - b|$.

It follows that $|\hat{\mu}_{j-1} - \theta_{j-1} + B_{j-1,n}| \leq |\hat{\mu}_{j-1} - \theta_{j-1}|$, and therefore (40) implies

$$|\hat{\mu}_j - \theta_j| \leq |\hat{\mu}_{j-1} - \theta_{j-1}| + |A_{j-1,n}| + |Z_j|. \tag{41}$$

Since $|\hat{\mu}_0 - \theta_0| = 0$ by definition, we have

$$|\hat{\mu}_R - \theta_R| \leq \sum_{j=1}^{R-1} |A_{j-1,n}| + \sum_{j=1}^{R} |Z_j|.$$

The desired bound in Proposition A.3 is then the consequence of two observations.

- Each $A_{j,n}$ is the difference between the sample mean of i.i.d. bounded random variables $\{C_\tau(Y_i - \theta_j)\}_{i \in [n]}$ and their expectation. $\mathbb{E}|A_{j-1,n}| \leq \sqrt{\mathbb{E} A_{j-1,n}^2} = O\left(\frac{\tau}{\sqrt{n}}\right)$.
- The Z_j's are independently drawn from $\mathcal{N}(0, \frac{4R\tau^2}{n^2 \rho})$. We have $E|Z_j| = O(\frac{\sqrt{R}\tau}{n\sqrt{\rho}})$.

F. Optimality of Boosting under Lossless Clipping

When the true data scales $\|\mathcal{X}\|, \|\mathcal{Y}\|$ are known, [4] studies the rate of convergence of AdaSSP estimator $\hat{\theta}_{\text{AdaSSP}}$ to the non-private least squares estimator θ^* by showing that, under mild regularity conditions for the design matrix X, the squared distance $\|\hat{\theta}_{\text{AdaSSP}} - \theta^*\|^2$ is less than $C\frac{\|\mathcal{X}\|^2 \text{tr}[(X^\top X)^{-2}]}{\epsilon^2/\log(1/\delta)}$ with probability at least $1-\delta/3$ (Theorem 3, [4]) for some constant C. As argued in the original paper, this rate of convergence is optimal for (ϵ, δ)-differentially private linear regression ([11], [24]).

In Theorem A.6 we show that the boosted algorithm can attain the same rate of convergence, which helps explain why BoostedAdaSSP performs no worse than one-shot AdaSSP in our experiments when the clipping threshold is data-dependent.

Theorem A.6. *If $y|X$ is sampled from a Gaussian linear model and the minimum eigenvalue of the Gram matrix satisfies $\lambda_{\min}(X^\top X) \geq \frac{\alpha n \|\mathcal{X}\|^2}{d}$ for some $\alpha > 0$, then there exists a high-probability event E not depending on $y|X$ such that, as long as the number of boosting rounds $T = O(1)$, the BoostedAdaSSP estimator $\hat{\theta}_T$ satisfies*

$$\mathbb{E}(\|\hat{\theta}_T - \theta^*\|^2 | X, E) \leq C\frac{\|\mathcal{X}\|^2 tr[(X^\top X)^{-2}]}{\epsilon^2/\log(1/\delta)} \tag{42}$$

for some constant C.

Proof. Let M denote the sum of $\hat{\lambda}I$ (where $\hat{\lambda}$ is defined in Algorithm 1) and the symmetric Gaussian matrix perturbation to $X^\top X$. convergence of Algorithmsider two convergence of Algorithmsecutive iterates of Algorithm 1. We have

$$\theta_{t+1} = (X^\top X + M)^{-1} X^\top (y - X\theta_t) + Z_{t+1}, \tag{43}$$

where $Z_{t+1} \sim \mathcal{N}_d\left(0, \nu^2 (X^\top X + M)^{-2}\right)$ is the fresh Gaussian noise drawn in the $(t+1)$-th iteration. The coefficient ν^2 in the covariance matrix depends on privacy parameters and data scales and shall be specified later.

With $\theta^* = (X^\top X)^{-1} X^\top y$, rearranging terms yields

$$\theta_{t+1} - \theta^* = (I - (X^\top X + M)^{-1} X^\top X)(\theta_t - \theta^*) + Z_{t+1}. \tag{44}$$

With the same choice of high-probability event as Section B of [4], we have $0.5(X^\top X + \hat{\lambda}I) \preceq X^\top X + M \preceq 2(X^\top X + \hat{\lambda}I)$, and therefore there exists some absolute convergence of Algorithmstant $0 < \kappa < 1$ such that

$$\|\theta_{t+1} - \theta^*\| \leq (1-\kappa)\|\theta_t - \theta^*\| + \|Z_{t+1}\|. \tag{45}$$

Iterating this noisy convergence of Algorithmtraction yields

$$\|\theta_{t+1} - \theta^*\| \leq (1-\kappa)^t \|\theta_1 - \theta^*\| + \sum_{j=1}^{t} (1-\kappa)^{t-j} \|Z_j\|. \tag{46}$$

To bound the right side in expectation, observe that $\mathbb{E}(\|\theta_1 - \theta^*\|^2 | X, E)$ is of the same order as $\mathbb{E}\|\hat{\theta}_{\text{AdaSSP}} - \theta^*\|^2 | X, E$ when $T = O(1)$. For the noise term, again with $T = O(1)$, the requisite ν^2 in AdaSSP is of the order $\frac{\|\mathcal{X}\|\|\mathcal{Y}\|}{\epsilon^2/\log(1/\delta)}$; the overall magnitude of the noise term $\frac{\|\mathcal{X}\|^2\|\mathcal{Y}\|^2}{\epsilon^2/\log(1/\delta)} \text{tr}[(X^\top X + M)^{-2}]$ is bounded by $C\frac{\|\mathcal{X}\|^2 \text{tr}[(X^\top X)^{-2}]}{\epsilon^2/\log(1/\delta)}$ under the high probability event for $(X^\top X + M)^{-1}$ and with the same choice of C in Theorem 2(iii) of [4]. □

G. Additional Theoretical Perspectives & Robustness of BoostedAdaSSP

In this section, we provide some additional theoretical justification of BoostedAdaSSP.

- In Section G1, we will prove a more fine-grained finite sample separation result between one-shot AdaSSP and BoostedAdaSSP: even if we know a bounded support of the data a priori at $[-B, B]$, and even if we can choose the threshold τ optimally as a function of this bound B and the sample size n, BoostedAdaSSP can already adapt to the small-variance of the actual data distribution with $T = 2$ steps, while non-BoostedAdaSSP with an optimally chosen τ cannot do any better than the worst-case that depends on the global boundedness parameter B.
- In Section G2, we derive a new interpretation of BoostedAdaSSP as an iterative optimization algorithm optimizing a "robustified" objective function from a fixed-point iteration perspective. This analysis offers new theoretical insight into the practical benefits of choosing large T and small τ.

Throughout this section, $[x]_{[a,b]} = \min(\max(x, b), a)$ denotes the clipping operator. To avoid the notational collision with the μ-Gaussian Differential Privacy, we use define $\rho := \mu^2$ and will use ρ for the privacy parameter throughout the section. This can be interpreted as ρ-zCDP or $\sqrt{\rho}$-GDP. The symbol μ will reserved for the mean of the random variable.

1) Finite-Sample Separation between AdaSSP and BoostedAdaSSP: To see the constant error incurred by AdaSSP in the finite-sample setting, consider estimating $\mu = \mathbb{E}Y$ using a private data set $Y_1, ..., Y_n \sim \mathcal{P}$. The worst-case MSE of one shot AdaSSP is characterized by the following theorem.

Theorem A.7. *Suppose there exists parameter B, σ, μ, such that the distribution $\mathcal{P}$ of random variable $Y \in \mathbb{R}$ satisfies that $\Pr(|Y| \leq B) = 1$, (b) $\mu = \mathbb{E}Y$, (c) $Y - \mu$ is σ^2-subgaussian. Let $\hat{\mu}_n^\tau(Y)$ be the AdaSSP estimator with clipping at τ. If its zCDP parameter $\rho < n$ and $n \geq C\sqrt{\log n/\rho}$ for a universal constant C, then for any clipping level τ and any n, we have*

$$\max_{\mathcal{P}} \mathbb{E}[(\hat{\mu}^\tau(Y) - \bar{\mu})^2] \geq \frac{\tau^2 + \mu^2}{18n^2\rho} + \frac{2\max^2(B - \tau, 0)}{9} \tag{47}$$

In addition, there exists parameters τ_1, τ_2 such that BoostedAdaSSP for two iterations with threshold in first round chosen as τ_1 and second round chosen as τ_2 such that

$$\mathbb{E}[(\hat{\mu}_2^{\tau_{1:2}}(Y) - \bar{\mu})^2] = O\left(\frac{\sigma^2 \log(n\rho) + \mu^2}{n^2\rho} + \frac{B^2 + \mu^2}{n^4\rho^2}\right). \tag{48}$$

Theorem A.7 implies that the one-step AdaSSP cannot gain by choosing $\tau < B$ without incurring a *non-vanishing* constant asymptotic error. Moreover, for large n, a dependence on B in the leading term is necessary no matter how τ is chosen.

Meanwhile, BoostedAdaSSP with $T = 2$ is able to get rid of the dependence in B from the leading term by adaptively choosing the clipping threshold. This separation can be orders-of-magnitude when $B \gg \max\{\sigma, \mu\}$.

We further note that the expression of interest we consider is how well a differentially private estimator can approximate the empirical mean $\bar{\mu}$. Results for estimating the population level mean parameter μ are directly implied, since $\mathbb{E}[(\bar{\mu} - \mu)^2] \leq \frac{\sigma^2}{n}$. Whenever ρ is small, or $\mu \gg \sigma$ is large, or $B \gg q$, the additional error due to DP from (47) and (48) could easily become larger than the statistical error for small to moderate sized data. From this perspective, Boosting in AdaSSP could make a difference in enabling applications of private data analysis to significantly smaller datasets than its non-boosted counterpart.

Proof of Theorem A.7. Let the noise added be Z_1, Z_2 from the first round of AdaSSP. Specifically, Z_1 is added to n, and Z_2 is added to $\sum_i (Y_i)_{[-\tau,\tau]}$. Z_3 is added to $\sum_i (Y_i - \hat{\mu}_1)_{[-\tau_2,\tau_2]}$.

Condition on the event E_1 that $|Z_1| \leq \sigma_1\sqrt{2\log(2/\delta)}$, which happens with probability $\geq 1 - \delta$ by the standard Gaussian tail bound. Under the assumption $n > 2\sigma_1\sqrt{2\log(2/\delta)}$ (we will work out the choice of σ_1 and δ later such that this will match what's stated in the theorem), this implies that $|Z_1| \leq n/2$. Observe that the conditional random variable $Z_1|E_1$ remains σ^2-subgaussian with a variance at least $0.5\sigma^2$. [4] Also observe that, under the same event and assumption on n, the clipping in the denominator at 1 does not occur, i.e., $\max\{1, n + Z_1\} = n + Z_1$.

[4] This can be checked by the variance of truncated Gaussian random variable.

Thus under E_1, we can write

$$\begin{aligned}\hat{\mu}^\tau(Y) - \bar{\mu} &= \frac{\sum_i [Y_i]_{[-\tau,\tau]} + Z_2}{n + Z_1} - \frac{\sum_i Y_i}{n} \\ &= \frac{\sum_i [Y_i]_{[-\tau,\tau]} + Z_2}{n + Z_1} - \frac{\sum_i Y_i}{n + Z_1} + \frac{\sum_i Y_i}{n + Z_1} - \frac{\sum_i Y_i}{n} \\ &= \frac{\sum_i [Y_i]_{[-\tau,\tau]} - Y_i}{n + Z_1} \frac{Z_2}{n + Z_1} + \frac{-Z_1 \sum_i Y_i}{n(n + Z_1)} \\ &= \underbrace{\frac{\sum_i [Y_i]_{[-\tau,\tau]} - Y_i}{n + Z_1}}_{(*)} + \underbrace{\frac{Z_2 - Z_1\bar{\mu}}{n + Z_1}}_{(**)}. \end{aligned} \tag{49}$$

Next, we discuss two cases. First consider $\tau \geq B$, the first term $(*)$ is 0 and it remains to bound the second term $(**)$

$$\left|\frac{(Z_2 - Z_1\bar{\mu})}{3n/2}\right|^2 \leq |\hat{\mu}^\tau - \bar{\mu}|^2 \leq \left|\frac{(Z_2 - Z_1\bar{\mu})}{n/2}\right|^2 \tag{50}$$

where conditioning on $Y_{1:n}$ and E_1, $\frac{(Z_2 - Z_1\bar{\mu})}{n/2}$ is subgaussian with parameters $\frac{\sigma_2^2 + \bar{\mu}^2\sigma_1^2}{(n/2)^2}$. Note that $\sigma_2^2 = \frac{\tau^2}{\rho}$ and $\sigma_1^2 = \frac{1}{\rho}$ (for an even splitting of the privacy budget). Taking conditional expectation gives

$$\frac{4(\tau^2 + \bar{\mu}^2/4)}{9n^2\rho} \leq \mathbb{E}[(\hat{\mu}^\tau - \bar{\mu})^2 | \bar{\mu}, E_1] \leq \frac{4(\tau^2 + \bar{\mu}^2)}{n^2\rho}. \tag{51}$$

Take expectation over on both sides over $\bar{\mu}|E_1$ (notice that $\bar{\mu}$ and E_1 are independent) we obtain an upper and bound of the form $\frac{C(\tau^2 + \mu^2 + \mathrm{Var}(\bar{\mu}))}{n^2\rho}$ for constant $C = 4$ and $C = 1/9$ respectively.

By Assumption (c), $\mathrm{Var}(\bar{\mu}) \leq \sigma^2/n$. This gives rise to an upper bound

$$\mathbb{E}[(\hat{\mu} - \bar{\mu})^2 | E_1] \leq \frac{4(\tau^2 + \mu^2)}{n^2\rho} + \frac{4\sigma^2}{n^3\rho}. \tag{52}$$

Note that under E_1^c, we have

$$\mathbb{E}[(\hat{\mu}^\tau - \bar{\mu})^2 | E_1^c] = \mathbb{E}[(\sum_i [Y_i]_{[-\tau,\tau]} + Z_2 - \frac{\sum_i Y_i}{n})^2 | E_1^c] \leq n\tau^2 + \frac{\tau^2}{\rho} + \mu^2 + \frac{\sigma^2}{n}. \tag{53}$$

It follows that if we choose $\delta \leq 1/n^3$ (under the assumption $\rho < n$)

$$\mathbb{E}[(\hat{\mu} - \bar{\mu})^2] \leq (1 - \delta)\mathbb{E}[(\hat{\mu}^\tau - \bar{\mu})^2 | E_1] + \delta\mathbb{E}[(\hat{\mu}^\tau - \bar{\mu})^2 | E_1^c] = O(\frac{(\tau^2 + \mu^2 + \sigma^2/n)}{n^2\rho}), \tag{54}$$

Moreover, it is clear that by a subgaussian concentration bound, we can prove a high probability error bound (in the $\tau \geq B$ regime) which says that with probability $1 - 2\delta$,

$$(\hat{\mu}^\tau - \bar{\mu})^2 \leq \frac{8(\tau^2 + \mu^2\sigma^2/n)\log(4/\delta)}{n^2\rho}. \tag{55}$$

To get a matching lower bound, consider a particular $\mathcal{P}$ such that $\Pr(Y = \mu) = 1$, thus $\mathrm{Var}(\bar{\mu}) \geq 0$ and

$$\max_{\mathcal{P}} \mathbb{E}[(\hat{\mu}^\tau(Y) - \bar{\mu})^2] \geq \frac{(1 - \delta)(\tau^2 + \mu^2)}{9n^2\rho}. \tag{56}$$

Now let's consider the case when $\tau < B$. We can still start from (49). Observe that we can still use (51) to bound $(**)$, in fact, we can obtain the same $\mathbb{E}[(**)^2] \geq \frac{(1-\delta)(\tau^2+\mu^2)}{9n^2\rho}$ for any $\tau < B$ too. It remains to construct a lower bound $(*)$ for using the same family of distributions. The idea is that $(*)$ will introduce additional bias that is not vanishing even if $n \to \infty$.

Again, we will consider a trivial distribution where $\Pr(Y = \mu) = 1$. Assume $\mu > \tau$, then

$$\max_{\mathcal{P}} \mathbb{E}\left[\left|\frac{\sum_i [Y_i]_{[-\tau,\tau]} - Y_i}{n + Z_1}\right|^2\right] \geq \max_{\mu > \tau} \mathbb{E}[(\frac{n(\mu - \tau)_+}{n + Z_1})^2] \tag{57}$$

$$\geq \max_{\mu > \tau} \Pr(E_1)(\frac{n(\mu - \tau)_+}{3n/2})^2 = \frac{4(1 - \delta)\max^2(B - \tau, 0)}{9}, \tag{58}$$

where we applied the $|Z_1| \leq n/2$ which is implied by E_1, for which we will choose $\delta < 1/2$.

Clearly, the above lower bound says that if we stick with $T = 1$, one cannot gain anything by choosing $\tau < B$ in terms of the max error.

Next, we will analyze the algorithm with $T = 2$. We will set $\tau_1 = B$ and $\tau_2 = O\left(\max\{\frac{B}{n\sqrt{\rho}}, \sigma\}\sqrt{\log(n)}\right)$.

The second boosting round will estimate

$$\hat{\mu}_2 = \hat{\mu}_1 + \frac{\sum_i [Y_i - \hat{\mu}_1]_{[-\tau_2,\tau_2]} + Z_3}{n + Z_1}. \tag{59}$$

where $Z_3 \sim \mathcal{N}(0, \sigma_3^2)$

Besides E_1, we will further consider the following high probability events.

(E_2) Following (55),with probability $1 - 2\delta$, $|\hat{\mu}_1 - \bar{\mu}| \leq \frac{CB\sqrt{\log(4/\delta)}}{n\sqrt{\rho}}$.

(E_3) With probability $1 - \delta$ for all $i = 1, ..., n$, $|Y_i - \mu| \leq \sigma\sqrt{2\log(2n/\delta)}$.

(E_4)Again by subgaussian concentration, with probability $1 - \delta$, $|\bar{\mu} - \mu| \leq \sigma\sqrt{2\log(2/\delta)}$.

Thus with by triangle inequality, with probability $1 - 4\delta$, $|\hat{\mu}_1 - Y_i| \leq \max\{\frac{B}{n\sqrt{\rho}}, \sigma\}\sqrt{C\log(2n/\delta)}$ for a constant C.

Let event $E = E_1 \cap E_2 \cap E_3 \cap E_4$. Under event E, when τ_2 is set with such a bound, clipping will not happen for any data point and

$$\begin{aligned}
\hat{\mu}_2 - \bar{\mu} &= \hat{\mu}_1 - \frac{\sum_{i=1}^n(-\hat{\mu}_1)}{n + Z_1} + \frac{\sum_i Y_i + Z_3}{n + Z_1} - \bar{\mu} \\
&= \frac{Z_1\hat{\mu}_1}{n + Z_1} + \frac{Z_3 + Z_1\bar{\mu}}{n + Z_1} \\
&= \frac{Z_1(\hat{\mu}_1 - \bar{\mu})}{n + Z_1} + \frac{Z_3 + 2Z_1\bar{\mu}}{n + Z_1}
\end{aligned} \tag{60}$$

It follows that

$$\mathbb{E}\left[(\hat{\mu}_2 - \bar{\mu})^2 | E\right] \leq 2\mathbb{E}\left[(\frac{Z_1(\hat{\mu}_1 - \bar{\mu})}{n + Z_1})^2 \middle| E\right] + 2\mathbb{E}\left[(\frac{Z_3 + 2Z_1\bar{\mu}}{n + Z_1})^2 \middle| E\right] \tag{61}$$

$$= O\left(\frac{B^2 + \mu^2}{n^4\rho^2} + \frac{\sigma^2\log(n/\delta) + \mu^2}{n^2\rho}\right). \tag{62}$$

Under the complement event E^c, we apply a trivial bound that can be enforced by the algorithm (clipping the estimate at $[-B, B]$ knowing that $\mu \in [-B, B]$)

$$\mathbb{E}[(\hat{\mu}_2 - \bar{\mu})^2 | E^c] = 4B^2. \tag{63}$$

Choose $\delta = \frac{1}{n^4\rho^2}$, by the law of total expectation, we get

$$\mathbb{E}\left[(\hat{\mu}_2 - \bar{\mu})^2\right] = O\left(\frac{B^2 + \mu^2}{n^4\rho^2} + \frac{\sigma^2\log(n\rho) + \mu^2}{n^2\rho}\right), \tag{64}$$

which completes the proof. □

2) Robustness to outliers via clipping and boosting — a fixed-point iteration persepctive: An additional benefit of boosting is robustness to outliers. For simplicity, here we analyze the non-private version of BoostedAdaSSP without noise addition. The reason for this simplification is that the robustness of BoostedAdaSSP can be revealed even without noise. Adding noise for privacy entails straightforward modification to the analysis below.

We consider a contaminated dataset with n data points $Y_1, ..., Y_n$. Among them $n - k$ are in-liers drawn i.i.d. from $\mathcal{P}$ - an arbitrary distribution supported on $[\mu - \sigma, \mu + \sigma]$ with mean μ, and k are outliers with a value of B. Without loss of generality, we assume the in-liers are the first $n - k$.

A non-boosted algorithm will have to add noise proportional to B to the non-private estimator given by $\hat{\mu} = \frac{kB + \sum_{i=1}^{n-k} Y_i}{n}$. The MSE for estimating $\mu := \mathbb{E}_{\mathcal{P}}Y$ is

$$\mathbb{E}(\hat{\mu} - \mu)^2 = \frac{k^2(B - \mu)^2}{n^2} + \frac{\sigma^2(n - k)}{2n^2}. \tag{65}$$

Note that $|B - \mu|$ can be arbitrarily large. The estimator is thus vulnerable to outliers.

For the boosted version of the algorithm, we fix a relatively small clipping threshold τ through many iterations until convergence. The solution can be viewed as iteratively solving the nonlinear equation

$$\sum_{i=1}^n (Y_i - \mu)_{[-\tau,\tau]} = 0, \tag{66}$$

because algorithmically, BoostedAdaSSP iteratively simulates the following fixed point equation until convergence:

$$\hat{\mu}_{t+1} = \hat{\mu}_t + \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{\mu}_t)_{[-\tau,\tau]}. \tag{67}$$

Theorem A.8 explains why solution to (66), to which the noiseless BoostedAdaSSP algorithm converges, is robust to outliers.

Theorem A.8. *As $\tau \to 0$, the solution to (66) converges to $\mathrm{Median}(Y_{1:n})$. For a constant τ, the solution to (66) minimizes the Huber loss with radius τ.*

Proof of Theorem A.8. When $\tau \to 0$, we may rewrite the equation as $\lim_{\tau\to 0}\frac{1}{\tau}\sum_{i=1}^{n}(Y_i-\mu)_{[-\tau,\tau]} = 0$. Then, observe that $\lim_{\tau\to 0}\frac{1}{\tau}(\mu - Y_i)_{[-\tau,\tau]} \in \partial|\mu - Y_i|$, and a valid solution certifying the sub-gradient optimality condition for $\min_\mu \sum_i |Y_i - \mu|$ is the sample median. For the second statement, observe that $(\mu - Y_i)_{[-\tau,\tau]} = \frac{\partial}{\partial\mu}\mathrm{Huber}_\tau(\mu - Y_i)$. A fixed point satisfying the optimality condition is a minimizer. □

Minimizers of the Huber loss functions is among the most well-known robust M-estimators studied in the classical literature. In the regime when $\hat{\mu}_t$ is τ-away from the true support, the iteration moves towards μ as long as the outlier is less than half of the data. Assume $2\sigma < \tau < B - \mu$, once $\hat{\mu}_t \in [\mu - \tau, \mu + \tau]$ the in-lier data will no longer be clipped, and the fixed point equation becomes $\sum_{i=1}^{n-k}(Y_i - \hat{\mu}) + k(B - \hat{\mu})_{[-\tau,\tau]} = 0$, which gives $\hat{\mu}_\infty = \frac{\sum_{i=1}^{n-k} Y_i + k(B-\hat{\mu}_\infty)_{[-\tau,\tau]}}{n-k}$, with an MSE error bound of $\mathbb{E}(\hat{\mu}_\infty - \mu)^2 \le \frac{2k^2\tau^2}{(n-k)^2} + \frac{2\sigma^2}{(n-k)}$.

Compared to the error of empirical mean estimator (65), the BoostedAdaSSP with a moderate τ and large T converges to a solution that is significantly more robust to outliers.