Differentially Private Bayesian Linear Regression

Naysan Saran, Joseph Antoun and Mohamed Amine Arfaoui

***Abstract*— Linear regression is an important tool across many fields that work with sensitive human-sourced data. Significant prior work has focused on producing differentially private point estimates, which provide a privacy guarantee to individuals while still allowing modelers to draw insights from data by estimating regression coefficients. The problem of Bayesian linear regression, with the goal of computing posterior distributions that correctly quantify uncertainty given privately released statistics, was investigated in [1]. In the aforementioned paper, it was shown that a naive approach that ignores the noise injected by the privacy mechanism does a poor job in realistic data settings. This has motivated the corresponding authors to develop noise-aware methods that perform inference over the privacy mechanism and produce correct posteriors across a wide range of scenarios. In this paper, the methods developed in [1] are first reproduced and then enhanced through various experiments. Finally, based on the outcome of the present paper and the current state of the art, some future research directions are proposed.**

***Keywords*—Logistic regression, LDA, cross validation, accuracy, complexity, features selection, features construction.**

I. Introduction

Linear regression is one of the most widely used statistical methods, especially in social sciences [1], [2] and other domains where data comes from humans. It is important to develop robust tools that can realize the benefits of regression analyses while maintaining the privacy of individuals. On the other hand, differential privacy [3] is a widely known paradigm that provides algorithmic privacy. In fact, a differentially private algorithm randomizes its computation to provably limit the risk that the outputs disclose information about individuals. Existing works on differentially private linear regression focus on frequentist approaches. A variety of privacy mechanisms have been applied to the estimation of regression coefficients, including sufficient statistic perturbation (SSP) [4]–[8], posterior sampling (PS) [7]–[12], subsample and aggregate [13], [14], objective perturbation (OP) [15], and noisy stochastic gradient descent (SGD) [16]. for the same problem, only a few recent works address uncertainty quantification through confidence interval estimation [17] and hypothesis tests [18].

A Bayesian approach for linear regression naturally quantifies parameters' uncertainty through a full posterior distribution and provides other Bayesian capabilities such as the ability to incorporate prior knowledge and compute posterior predictive distributions. Existing approaches to private Bayesian inference include PS, Markov chain Monte Carlo (MCMC) [12], variational inference (VI) [19]–[21], and SSP [4], [22], but none provide a fully satisfactory approach for Bayesian regression modeling. PS does not naturally produce a representation of a full posterior distribution. MCMC approaches incur per-iteration privacy costs and satisfy only approximate (ϵ, δ)-differential privacy. Private VI approaches also incur per-iteration privacy costs, and are most relevant when the original inference problem requires VI. When applicable, SSP is a very desirable approach, since sufficient statistics are perturbed once and then used in conjugate updates to obtain parameters of full posterior distributions, and often outperforms other methods in practice [4], [7]. In [22] it was demonstrated that, for unconditional exponential family models, naive SSP, which ignores noise introduced by the privacy mechanism, systematically underestimates uncertainty at small to moderate sample sizes. In [1], a differentially private method for Bayesian linear regression was proposed and it was shown that the same phenomenon holds for Bayesian linear regression. Specifically, naive SSP produces private posteriors that are properly calibrated asymptotically in the sample size, but for realistic data sets and privacy levels may need very large population sizes to reach the asymptotic regime.

These results and observations motivated the authors in [1] to develop Bayesian inference methods for linear regression that properly account for the noise due to the privacy mechanism. Precisely, in [1], a model, in which data and model parameters are latent variables, and noisy sufficient statistics are observed, was leveraged first. Then, some MCMC-based techniques to sample from posterior distributions were developed, as done for exponential families [22]. A significant challenge relative to prior works is the handling of covariate data. Typical regression modeling treats only response variables and parameters as random, and conditions on covariates. However, this is not possible in the private setting, since covariates must be kept private and therefore treated as latent variables. To overcome this issue, some assumptions about the distribution over covariates were required. Based on this, two inference methods were developed. The first includes latent variables for each individual, i.e., it requires an explicit prior distribution for covariates. The second marginalizes out individuals and approximates the distribution over the sufficient statistics, i.e., it requires weaker assumptions about the covariate distribution (only moments). One of the main differences between the two methods is that the runtime of the first method scales with the population size, whereas the ones of the second method does not. Finally, it was shown that, through various experiments real data, the noise-aware methods are as well or nearly as well calibrated as the non-private method, and have better utility than the naive method and that they quantify posterior predictive uncertainty significantly better than naive SSP.

In this paper, we reproduce the Bayesian inference methods developed in [1]. Specifically, we revise all the analytical analysis derived for the methods and we run various experi-

Naysan Saran is within The Department of Mathematics and Statistics, McGill University, Montreal, Canada, (Email: naysan.saran@mail.mcgill.ca). Joseph Antoun and Mohamed Amine Arfaoui are within the Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montreal, Canada, (Emails: {j_antoun, m_arfaou}@encs.concordia.ca).

ments to reproduce the claimed experimental results. Our main contributions in this paper are listed as follows.

1) Adding two loops to recreate Figure 3.(a) and Figure 3.(d). The original code [23] does not have a loop in *driver.py* that displays these figures.
2) Adding a custom implementation of the Kolmogorov-Smirnov (KS) statistic using the *ks_2samp()* function of the *scipy* library.
3) Adding a custom implementation of the maximum mean discrepancy (MMD) metric, where we implemented a function that reproduces the expression $MMD^2(P,Q)$ that is shown in page 8 [1] and a function that emulates a 2-dimensional (2d) standard normal kernel to emulate the function $k(p,q)$ that is used for $MMD^2(P,Q)$.
4) Adding scripts to save and plot the values of the parameters and the posterior distributions for each Bayesian inference methods.
5) Improving the function *run_methods()* to keep track of the runtimes of the Bayesian inference methods.
6) Adding an experiment to test if the Noise-naive method produces an asymptotically correct posterior. The output result is displayed by plotting the mean posterior estimates of non-private method versus naive method.

The rest of the paper is organized as follows. Section II presents the analytical formulation of the Bayesian inference methods developed in [1]. Section III presents the implementation details of the Bayesian inference methods. Section IV presents the ablation studies and the proposed improvements. Section V presents the performance evaluation of the Bayesian inference methods along with the proposed improvements. Section VI presents the conclusion and some future research directions.

II. Bayesian Linear Regression

A. Non-Private Bayesian Linear Regression

Let $\mathbf{x} \in \mathbb{R}^d$ be an individual's covariate or regressor data and let $y \in \mathbb{R}$ be its dependent response data. Assuming a conditionally Gaussian model $y \sim \mathcal{N}\left(\boldsymbol{\theta}^T\mathbf{x}, \sigma^2\right)$, where $\boldsymbol{\theta} \in \mathbb{R}^d$ are the regression coefficients and σ^2 is the error variance, the goal behind linear regression is obtaining a point estimate of $\boldsymbol{\theta}$, given an observed population of n individuals. The population data can be written as $\mathbf{X} \in \mathbb{R}^{n\times d}$, where each row corresponds to an individual $\mathbf{x}$, and $\mathbf{y} \in \mathbb{R}^n$. Based on the results of [24], the ordinary least squares (OLS) solution is $\widehat{\boldsymbol{\theta}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$.

In Bayesian linear regression, the parameters $\boldsymbol{\theta}$ and σ^2 are random variables with a specified prior distribution. The conjugate priors are $p\left(\sigma^2\right) = \mathrm{IG}\left(a_0, b_0\right)$ and $p\left(\boldsymbol{\theta}|\sigma^2\right) = \mathcal{N}\left(\boldsymbol{\mu}_0, \sigma^2\boldsymbol{\Lambda}_0^{-1}\right)$, where IG refers to the inverse Gaussian distribution. This defines a normal-inverse gamma prior distribution $p\left(\boldsymbol{\theta}, \sigma^2\right) = \mathrm{NIG}\left(\boldsymbol{\mu}_0, \boldsymbol{\Lambda}_0, a_0, b_0\right)$. Due to conjugacy of the prior distribution with the likelihood model, the posterior distribution $p\left(\boldsymbol{\theta}, \sigma^2|\mathbf{X}, \mathbf{y}\right)$ is expressed as

$$p\left(\boldsymbol{\theta}, \sigma^2|\mathbf{X}, \mathbf{y}\right) = NIG\left(\boldsymbol{\mu}_n, \boldsymbol{\Lambda}_n, a_n, b_n\right), \tag{1a}$$

$$\boldsymbol{\Lambda}_n = \mathbf{X}^T\mathbf{X} + \boldsymbol{\Lambda}_0, \tag{1b}$$

$$\boldsymbol{\mu}_n = \boldsymbol{\Lambda}_n^{-1}\left(\mathbf{X}^T\mathbf{y} + \boldsymbol{\Lambda}_0^T\boldsymbol{\mu}_0\right), \tag{1c}$$

$$a_n = a_0 + \frac{n}{2}, \tag{1d}$$

$$b_n = b_0 + \frac{1}{2}\left(\mathbf{y}^T\mathbf{y} + \boldsymbol{\mu}_0^T\boldsymbol{\mu}_0\boldsymbol{\mu}_0 - \boldsymbol{\mu}_n^T\boldsymbol{\mu}_n\boldsymbol{\mu}_n\right), \tag{1e}$$

which is also a normal-inverse gamma distribution [25]. Additionally, the statistics

$$\mathbf{s} := t\left(\mathbf{X}, \mathbf{y}\right) = \sum_{i=1}^{n} t\left(\mathbf{x}_i, y_i\right) = \left[\mathbf{X}^T\mathbf{X}, \mathbf{X}^T\mathbf{y}, \mathbf{y}^T\mathbf{y}\right], \tag{2}$$

are sufficient statistics for the model in (1). These statistics capture all information about the model parameters contained in the sample $(\mathbf{X}, \mathbf{y})$ and are the only quantities needed for the conjugate posterior updates in (1) [26].

B. Private Bayesian Linear Regression

As opposed to the non-private case, the goal behind the private Bayesian linear regression is perform the same model but with an ϵ-differentially private manner. The privacy is ensured by employing sufficient statistic perturbation (SSP), in which the Laplace mechanism is used to inject noise into the sufficient statistics of the model, making them fit for public release [4], [6], [8]. The question is then how to compute the posterior over the model parameters $\boldsymbol{\theta}$ and σ^2 given the noisy sufficient statistics. To answer this question, a naive method that ignores the noise in the noisy sufficient statistics is considered first. We then more principled noise-aware inference approaches that account for the noise due to the privacy mechanism are studied.

1) Privacy Mechanism: Using the Laplace mechanism to release the noisy sufficient statistics $\mathbf{z}$ results in the model shown in Fig. 1. This is the same model used for non-private linear regression except the introduction of $\mathbf{z}$, which requires the exact sufficient statistics $\mathbf{s}$ to have finite sensitivity. A standard assumption in literature is to assume $\mathbf{x}$ and y have known a priori lower and upper bounds, $(a_\mathbf{x}, b_\mathbf{x})$ and (a_y, b_y), with bound widths $w_\mathbf{x} = b_\mathbf{x} - a_\mathbf{x}$ (assuming for simplicity equal bounds for all covariate dimensions) and $w_y = b_y - a_y$, respectively [7], [17], [27], [28]. Then, one can reason about the worst case influence of an individual on each component of $\mathbf{s} = \left[\mathbf{X}^T\mathbf{X}, \mathbf{X}^T\mathbf{y}, \mathbf{y}^T\mathbf{y}\right]$. Thus, recalling that $s = \sum_{i=1}^{n} t\left(\mathbf{x}_i, y_i\right)$, it follows that

$$\left[\Delta_{\left(\mathbf{X}^T\mathbf{X}\right)_{j,k}}, \Delta_{\left(\mathbf{X}^T\mathbf{y}\right)_j}, \Delta_{y^2}\right] = [w_\mathbf{x}^2, w_\mathbf{x}w_y, w_y^2]. \tag{3}$$

The number of unique elements in $\mathbf{s}$ is $\left[\frac{d(d+1)}{2}, d, 1\right]$, so $\Delta_\mathbf{s} = w_\mathbf{x}^2\frac{d(d+1)}{2} + w_\mathbf{x}w_y d + w_y^2$. The noisy sufficient statistics fit for public release are $\mathbf{z} = \left[z_i \sim \mathrm{Lap}\left(s_i, \frac{\Delta_\mathbf{s}}{\epsilon}\right), s_i \in \mathbf{s}\right]$, where Lap refers to the Laplace distribution.

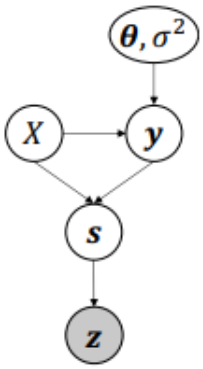

Fig. 1. Private regression model

2) Noise-Naive Method: Previous works developed methods to obtain OLS solutions via SSP by ignoring the noise injected into the sufficient statistics [7], [17], [27]. One corresponding approach for Bayesian regression is to naively replacing **s** in Fig. 1 with the noisy version **z** and then performing the conjugate update in (1). This noise-naive method is simple and fast, and it was empirically shown in [1] that it produces an asymptotically correct posterior.

3) Noise-Aware Inference: Instead of ignoring the noise introduced by the privacy mechanism as done in the previous paragraph, the authors in [1] proposed to perform inference over the noisy model in Fig. 1 in order to produce correct posteriors regardless of the data size. The biggest change from non-private to private Bayesian linear regression is that, due to privacy constraints, conditioning on the covariate data $\mathbf{X}$ is no longer possible. Based on [1], the non-private posterior probability is given by

$$p\left(\boldsymbol{\theta}, \sigma^2 | \mathbf{X}, \mathbf{y}\right) \propto p\left(\boldsymbol{\theta}, \sigma^2\right) p\left(\mathbf{y} | \mathbf{X}, \boldsymbol{\theta}, \sigma^2\right), \tag{4}$$

while the private posterior is given by

$$p\left(\boldsymbol{\theta}, \sigma^2 | \mathbf{z}\right) \propto \int p\left(\mathbf{X}\right) p\left(\boldsymbol{\theta}, \sigma^2\right) p\left(\mathbf{y} | \mathbf{X}, \boldsymbol{\theta}, \sigma^2\right) p\left(\mathbf{z} | \mathbf{X}, \mathbf{y}\right) \mathrm{d}\mathbf{X}\mathrm{d}\mathbf{y}. \tag{5}$$

The private posterior contains the term $p(\mathbf{X})$, which means that, in order to calculate it, it is required to know something about the distribution of $\mathbf{X}$. This approach is simple to implement using existing tools. However, it places a substantial burden on the model relative to the non-private case by requiring an explicit prior distribution $p(\mathbf{X})$, and hence, poor choices will potentially lead to incorrect inferences [1]. Additionally, since *MCMC-Ind* instantiates latent variables for each individual, its runtime scales with population size and it may be slow for large populations.

An appealing possibility to overcome the aforementioned issue is marginalizing out the variables $\mathbf{X}$ and $\mathbf{y}$ representing individuals and performing inference directly over the latent sufficient statistics $\mathbf{s}$ instead. Recall that the joint distribution is $p(\boldsymbol{\theta}, \sigma^2, \mathbf{s}, \mathbf{z}) = p(\boldsymbol{\theta}, \sigma^2) p(\mathbf{s} | \boldsymbol{\theta}, \sigma^2) p(\mathbf{z} | \mathbf{s})$. The goal is to compute a representation of $p(\boldsymbol{\theta}, \sigma^2 | \mathbf{z}) \propto \int p(\boldsymbol{\theta}, \sigma^2, \mathbf{s}, \mathbf{z}) d\mathbf{s}$ by integrating over the sufficient statistics. Due to the fact that the closed-form expression of this distribution is not readily available, the authors in [1] developed a Gibbs sampler to sample from the posterior distribution as done in their prior work for unconditional exponential family models [22] . This requires methods to sample from the conditional distributions for both the parameters $(\boldsymbol{\theta}, \sigma^2)$ and the sufficient statistics $\mathbf{s}$ given all other variables. The full conditional distribution $p(\boldsymbol{\theta}, \sigma^2 | \mathbf{s})$ for the model parameters can be computed and sampled using conjugacy, exactly as in the non-private case. The full conditional for $\mathbf{s}$ can be decomposed into two terms as $p(\mathbf{s} | \boldsymbol{\theta}, \sigma^2, \mathbf{z}) \propto p(\mathbf{s} | \boldsymbol{\theta}, \sigma^2) p(\mathbf{z} | \mathbf{s})$. The first is the distribution over sufficient statistics of the regression model, for which an asymptotically correct normal approximation was developed. The second is the noise model due to the privacy mechanism, for which variable augmentation is used to ensure the possibility to sample from the full conditional distribution of $\mathbf{s}$.

4) Normal Approximation for $p(\boldsymbol{s} | \boldsymbol{\theta}, \sigma^2)$: The conditional distribution over the sufficient statistics given the model parameters is expressed as

$$p(\mathbf{s} | \boldsymbol{\theta}, \sigma^2) = \int_{t^{-1}(\mathbf{s})} p(\mathbf{X}, \mathbf{y} | \boldsymbol{\theta}, \sigma^2) \mathrm{d}\mathbf{X}\mathrm{d}\mathbf{y}, \tag{6}$$

where

$$t^{-1}(\mathbf{s}) := \{\mathbf{X}, \mathbf{y} : t(\mathbf{X}, \mathbf{y}) = s\}. \tag{7}$$

The integral over $t^{-1}(\mathbf{s})$ (all possible populations which have sufficient statistic $\mathbf{s}$) is intractable to compute. Instead, note that the components of $\mathbf{s} = \sum_i t(x_i, y_i)$ are sums over individuals. Therefore, using the central limit theorem (CLT), their distribution can be approximated as $p(\mathbf{s} | \boldsymbol{\theta}, \sigma^2) \approx \mathcal{N}(\mathbf{s}; n\boldsymbol{\mu}_t, n\boldsymbol{\Sigma}_t)$, where $\boldsymbol{\mu}_t = \mathbb{E}[t(\mathbf{x}, y)]$ and $\boldsymbol{\Sigma}_t = \mathrm{Cov}(t(\mathbf{x}, y))$ are the mean and covariance of the function $t(\mathbf{x}, y)$ on a single individual. This approximation is asymptotically correct, i.e., $\frac{1}{\sqrt{n}}(\mathbf{s} - n\boldsymbol{\mu}_t) \xrightarrow{D} \mathcal{N}(0, \boldsymbol{\Sigma}_t)$ [29]. Hence, the conditional distribution can be approximated as $\mathbf{s}|\cdot \sim \mathcal{N}(n\boldsymbol{\mu}_t, n\boldsymbol{\Sigma}_t)$. The components of $\boldsymbol{\mu}_t$ and $\boldsymbol{\Sigma}_t$ can be written in terms of the model parameters $(\boldsymbol{\theta}, \sigma^2)$ and the second and fourth non-central moments of $\mathbf{x}$ [1]. This method is called *Gibbs-SS*. The current parameters values are available within the sampler, but the modeler must provide estimates for the moments of $\mathbf{x}$, either using prior knowledge or by privately estimating the moments from the data. To do so, three specific possibilities can be adopted, which are private sample moments (*Gibbs-SS-Noisy*), moments from generic prior (*Gibbs-SS-Prior*) and hierarchical normal prior (*Gibbs-SS-Update*). These three methods are presented in details in [1]. As it can be seen in the above analysis, more modeling assumptions are needed in the private case than in the non-private case, where it is possible to condition on $\mathbf{x}$. However, *Gibbs-SS* requires more assumptions (second and fourth moments) than MCMC-Ind (a full prior distribution).

5) Variable Augmentation for $p(\boldsymbol{z} | \boldsymbol{s})$: The above approximation for the distribution over sufficient statistics means of the full conditional distribution involves the product of a normal and a Laplace distribution as

$$p(\mathbf{s} | \boldsymbol{\theta}, \mathbf{z}) \propto \mathcal{N}(\mathbf{s}; n\boldsymbol{\mu}_t, n\boldsymbol{\Sigma}_t) \times \mathrm{Lap}(\mathbf{z}; \mathbf{s}, \frac{\Delta_{\mathbf{s}}}{\epsilon}). \tag{8}$$

It is unclear how to sample from this distribution directly. A similar situation arises in the Bayesian Lasso, where it is solved by variable augmentation [30]. In their prior work

in [22], Bernstein and Sheldon adapted the variable augmentation scheme to private inference in exponential family models. In [1], the same approach is adopted, where the authors represented a Laplace random variable as a scale mixture of normal distributions. Specifically, $l \sim \text{Lap}(u, b)$ is identically distributed to $l \sim \mathcal{N}(u, \omega^2)$ where the variance $w^2 \sim \text{Exp}(\frac{1}{2b^2})$ is drawn from the exponential distribution with density $\frac{1}{2b^2}\exp\left(-\frac{w^2}{2b^2}\right)$. Each component of the vector $\mathbf{z}$ was augmented separtely so that $\mathbf{z} \sim \mathcal{N}(\mathbf{s}, \text{diag}(\omega^2)$, where $\omega^2 \sim \text{Exp}\left(\frac{\epsilon^2}{2\Delta_{\mathbf{s}}^2}\right)$. The augmented full conditional $p(\mathbf{s}|\boldsymbol{\theta}, \mathbf{z}, \omega)$ is a product of two multivariate normal distributions, which is itself a multivariate normal distribution.

III. Implementation Details

In this section, we represent the implementations details of reproducing the Bayesian linear regression methods developed in [1].

A. Environment Settings

The characteristics of the machine on which all the simulations are performed are as follows.

- System Type: AWS EC2, m5ad.4xlarge.
- 16 CPU, 64 GiB RAM, 2 x 300 NVMe SSD

All the analysis was performed on Python 3.7.

B. Running the code

The first step to reproduce the results of [1] in this paper was to clone the associated GitHub repository and run the code. The main file is `driver.py`, which according to the GitHub `Readme`, generates toy data, calculates the posterior for each method, and plots them.

The code inside `driver.py` runs the following steps.

1) Initialize the number of individuals to be sampled $N = 1000$.
2) Initialize the privacy setting $\epsilon = 0.01$.
3) Initialize $data_{dim} = 1$, so we are effectively running a simple linear regression.
4) Call `setup_data()`, which returns the following variables
   - `data_prior_params` $[\mu', \Lambda', \Psi', \nu'] = [0, 1, 1, 50]$.
   - `model_prior_params` $[\mu, \Lambda, \alpha, \beta] = [[0, 0], \text{diag}([\frac{.5}{20-1}, \frac{.5}{20-1}]), 20, .5]$.
   - `true_params` $(\Theta, \sigma^2) \sim \text{NIG}(\mu, \Lambda, \alpha, \beta)$, $(\mu_x, \tau^2) \sim \text{NIW}(\mu', \Lambda', \Psi', \nu')$.
   - $\mathbf{X}$ and y, samples of size N generated using $\mathbf{X} \sim \mathcal{N}(\mu_{\mathbf{x}}, \tau^2)$ and $\mathbf{y} \sim \mathcal{N}(\mathbf{X}^T\boldsymbol{\theta}, \sigma)$
   - `sensitivity_x` $= x_{max} - x_{min}$
   - `sensitivity_y` $= y_{max} - y_{min}$
5) Call `run_methods()`, which returns 2000 samples from the posterior distribution for each of the following methods: 'non-private', 'naive, 'mcmc', 'gibbs-noisy' and 'gibbs-update'.

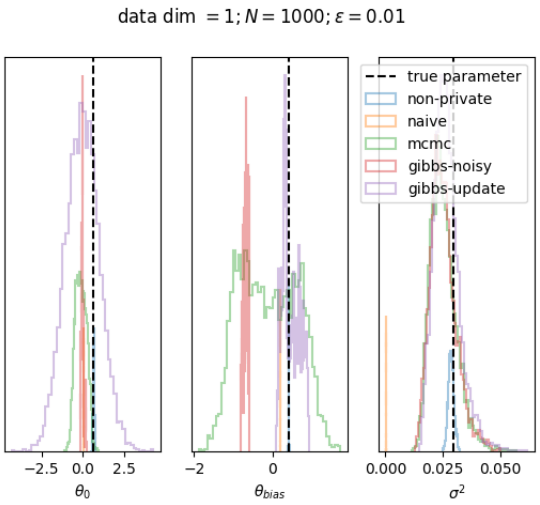

Fig. 2. Posterior distributions generated by driver.py

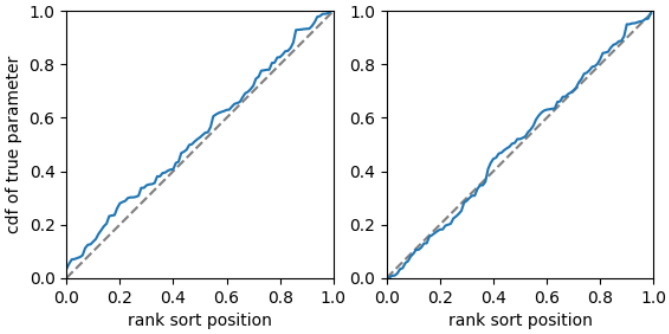

Fig. 3. Resulting QQ plot generated by NIW.py

6) Call `plot_posteriors()`, which plots the posterior distributions for each of the following parameters θ_0, θ_{bias} and σ_{squared}. The resulting plot is shown in Fig. 2.

Additionally, the file `NIW.py` contains a `__main__` routine that plots the quantile-quantile (QQ) plots of μ_x and τ for 100 trials of the following experiment

$$[\mu', \Lambda', \Psi', \nu'] = [0, 1, 1, 50],$$
$$(\mu_x, \tau^2) \sim NIW(\mu', \Lambda', \Psi', \nu'),$$
$$X \sim N(\mu_x, \tau^2),$$
$$S = X, X^T X,$$
$$posterior = ConjugateUpdate(S, [\mu', \Lambda', \Psi', \nu']).$$

which outputs the results in Fig. 3

C. Necessary Add-Ones

The paper displays several other figures that cannot be reproduced simply by running the code available in the GitHub repository. We have therefore created multiple new functions to perform the following tasks

1) For n in 10,100,1000, plot the $utility$ metric for θ_0 , θ_{bias} and $\sigma_{squared}$.
2) For n in 10,100,1000, plot the $calibration$ metric versus n for θ_0 , θ_{bias} and $\sigma_{squared}$, with $\epsilon = 0.1$.

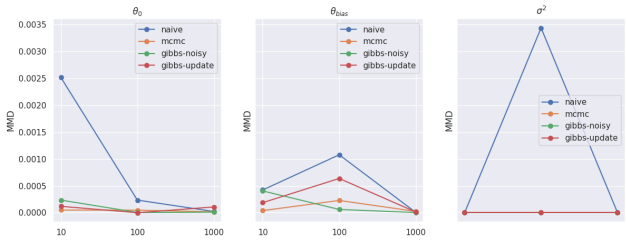

Fig. 4. Custom MMD plots for each parameter to be estimated, n=1000, epsilon=0.01

3) For ϵ in 0.01,0.1,1.0, plot the *calibration* metric versus ϵ for θ_0 , θ_{bias} and $\sigma_{squared}$, with $n = 10$.

1) Maximum Mean Discrepancy (MMD): The utility Maximum Mean Discrepancy (MMD) measures how close the computed posterior is to the non-private posterior $p(\theta, \sigma^2|s)$. In [1], the authors evaluated utility as *"closeness to the non-private posterior, which we measure with maximum mean discrepancy (MMD)"*. Unfortunately, we did not find any code to compute this metric in the GitHub repository associated to this paper. After trying several publicly available libraries with no success - they required theano tensors of dimension at least 2 but the posteriors generated in the experiments are one dimensional numpy arrays - we decided to implement our own functions to compute the maximum mean discrepancy. Our custom code is available inside `new_plots.py` and follows the formula from section 4.2 of the paper, which specifies that given m i.i.d samples $(p, q) \sim PxQ$, an unbiased estimate of the MMD is

$$\text{MMD}^2(P, Q) = \frac{1}{m(m-1)}\text{SUM}, \tag{9}$$

where

$$\text{SUM} = \sum_{i=j}^{m}(k(p_i, p_j)+k(p_i, p_j)-k(p_i, p_j)-k(p_i, p_j)). \tag{10}$$

Since the authors used the standard normal kernel for k, we used the following equation in our custom implementation

$$k(x, y; \sigma) = \frac{1}{2\pi\sigma^2}exp(-\frac{x^2+y^2}{2\sigma^2}), \tag{11}$$

where $\sigma = 1$. The available plots for $n = 1000$ and $\epsilon = 0.01$ are presented in Fig. 4. In order to create this figure, we looped through each the samples associated to the posterior distributions of each of the following methods 'naive, 'mcmc', 'gibbs-noisy' and 'gibbs-update' and computed their MMD score with the equivalent 'non-private' posterior sample. Furthermore, our custom MMD^2 function takes a long time to run for large values of m, so we had to limit ourselves to $m = 300$, effectively only computing the score for the first 300 samples of each distribution. Even if the shapes of the plots are different, we arrive to the same conclusion as the authors, namely that the 'Naive' method generates more distant (higher MMD, hence lower *utility*) posteriors.

```
#----------------------------------------
# Pseudo-Code to Generate the MMD Plot
#----------------------------------------
set epsilon = .01
methods     = [
'non-private', 'naive', 'mcmc',
'gibbs-noisy', 'gibbs-update'
]
for N in [10, 100, 1000]:
    setup_data()
    for method in methods:
        compute_posterior()
        save_posterior()
    for method in methods:
        if method != 'non_private':
            computeMMD(
                saved_posterior,
                non_private_posterior
            )
)
```

2) Kolmogorov-Smirnov (KS) statistic : The authors define *calibration* as the measure of how close the computed posterior is to $p(\theta, \sigma^2|z)$, the correct posterior given noisy statistics. To measure *calibration*, the paper chooses the Kolmogorov-Smirnov (KS) goodness-of-fit test [31]. The setup is as follows.

$$[\mu, \Lambda, \alpha, \beta] = [[0, 0], \text{diag}([\frac{.5}{20-1}, \frac{.5}{20-1}]), 20, .5],$$
$$[\mu', \Lambda', \Psi', \nu'] = [[0], [1], 1, 50],$$
$$(\Theta, \sigma^2) \sim \text{NIG}(\mu, \Lambda, \alpha, \beta),$$
$$\mu_x, \tau^2 \sim \text{NIW}(\mu', \Lambda', \Psi', \nu'),$$
$$X \sim N(\mu_x, \tau^2),$$

Again, neither the function to compute the KS-Statistic, nor the code to generate the calibration plots are available in the public GitHub repository, so we decided to implement a different version of the experiment and compare our conclusions to the ones from the paper. In this new experiment, we use the `ks_2samp()` function from `scipy.stats` which computes the Kolmogorov-Smirnov statistic on two samples. The pseudo-code is as follows.

```
#----------------------------------------
# Pseudo-Code to Generate the KS-stats
#----------------------------------------
methods = [
'non-private', 'naive', 'mcmc',
'gibbs-noisy', 'gibbs-update'
]
for N in [10, 100, 1000]:
    epsilon = .01
    setup_data()
    for method in methods:
        compute_posterior()
        save_posterior()
    for method in methods:
        if method != 'non_private':
            compute_KS_stat(
                saved_posterior,
                non_private_posterior
            )
)
```

The resulting plot is shown in Fig. 5. We also ran a similar logic, this time by setting $N = 10$ and looping through $\epsilon \in [0.01, 0.1, 1.0]$ as in Figure 3, plot (b) of the paper, resulting in Fig. 6. Interestingly, even with this different method, we arrive at the same conclusion as suggested by the equivalent figures from the paper, namely that the 'Naive' method always generates the most distant posterior distribution, regardless of

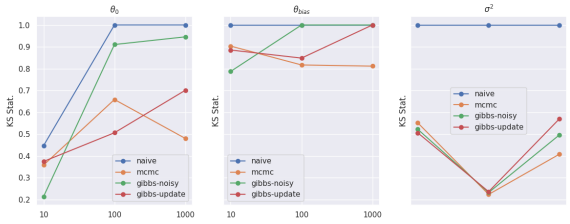

Fig. 5. Custom KS-statistics plots for n in 10,100,1000, epsilon=0.01

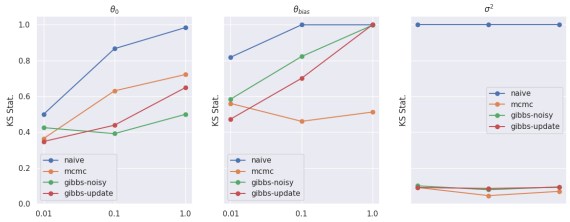

Fig. 6. Custom KS-statistics plots for n=10, epsilon in 0.01,0.1,1.0

the parameter. However, besides from the case of the parameter σ^2, our plots do not seem show the same strong clustering of the non-naive methods as the original paper.

IV. Ablation Studies and Improvements

In this paper, and compared to the results of [18], we represent the following improvements/ablation experiments.

1) Isolate and improve the process of generating the synthetic data, as well as the generation of the posterior distributions for each model
2) Test the statement from section 3.2 of the paper according to which the Noise-naive method produces an asymptotically correct posterior.

A. Isolating and improving the process of generating the synthetic data and the posterior distributions

Upon reading the paper and running the code, one of the first things that we noticed, was that there was no script available to plot the synthetic data that was generated by the function `setup_data()`. As a first improvement, we added the following 3 scripts

1) `experiment1_driver.py`
   sets $\epsilon = 0.01$ loops through $N = 10, 100, 1000$, and saves all the relevant data (true parameters, synthetic data, posterior distributions...). Another loop loads the saved files and plots them.
2) `experiment1_save_data.py`
   implements the functions `save_true_params()` and `save_posteriors()` that turn the variables into csv data-frames, and saves them as csv files inside the `./data` sub-directory.
3) `experiment1_plots.py`
   implements `plot_data_distributions()` that loads the csv data-frames inside the `./data` sub-directory and plots them. We also added a frequentist Ordinary Least Squares fit to the synthetic data for the sake of comparison.

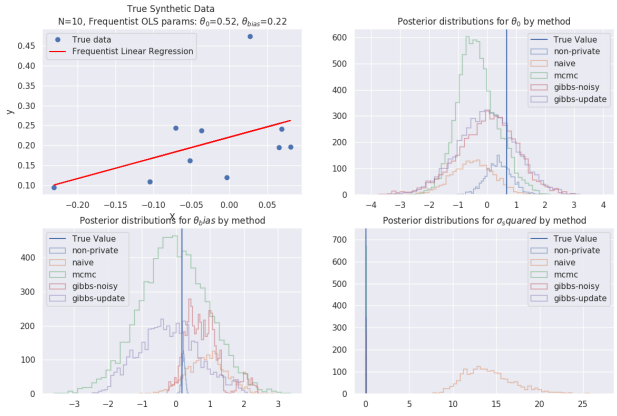

Fig. 7. Synthetic data and posterior distributions for n=10, epsilon=0.01

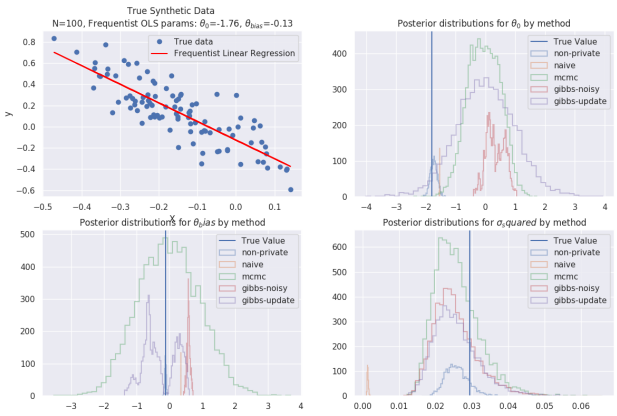

Fig. 8. Synthetic data and posterior distributions for n=100, epsilon=0.01

The results of this experiment are presented in Fig. 7 (n=10), Fig. 8 (n=100), and Fig. 9 (n=1000). We notice that, besides the case of σ^2, the precision of the non-naive methods do not necessarily improve as N increases. The case of the noise-aware MCMC method is interesting: as N increases, the posterior distributions of θ_0 and θ_{bias} display less variance, so in theory we should be more certain about the values of these parameters, yet their bias increases as the mean value of the posterior distributions become more distant to the true parameter.

B. Does the Noise-naive method produce an asymptotically correct posterior

In [18], the method that is used to add privacy to the data is adding Laplace noise to the sufficient statistics of the model $\mathbf{s} := t(\mathbf{X}, \mathbf{y})$, so that the noisy sufficient statistics fit for public

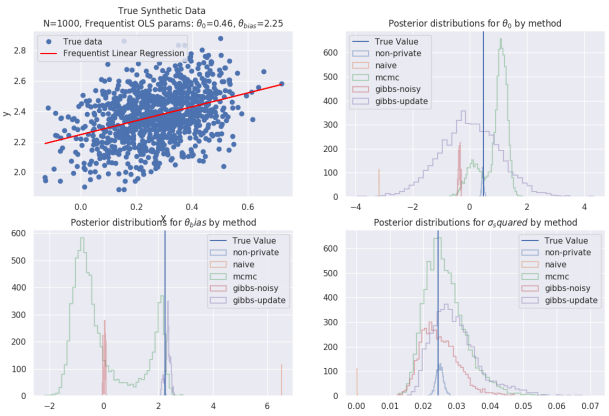

Fig. 9. Synthetic data and posterior distributions for n=1000, epsilon=0.01

Param	True Value
theta-0	-2.573476124657823
theta-bias	1.6327237063518738
sigma-squared	0.04339340003526078
mu-x	0.048364541952864874
tau	0.019778847315076873

TABLE I
CONTENTS OF ./DATA/TRUE-PARAMS-N1000-E0.01.CSV

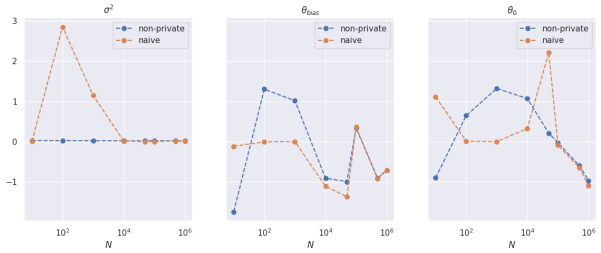

Fig. 10. Does the Noise-naive method produce an asymptotically correct posterior?

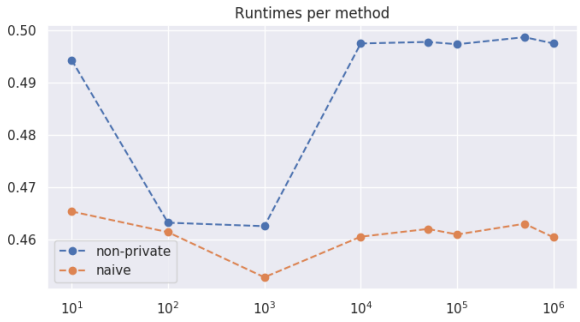

Fig. 11. Run-times per method, in seconds, for experiment 2

release are $\mathbf{z} = [z_i \sim \text{Lap}(s_i, \frac{\Delta_s}{\epsilon} | s_i \in s)]$. In the published code on GitHub [23], the experiment goes as follows:

- setup-data($datadim, N$)

$$[\mu, \Lambda, \alpha, \beta] = [[0,0], diag([\frac{.5}{20-1}, \frac{.5}{20-1}]), 20, .5],$$
$$[\mu', \Lambda', \Psi', \nu'] = [[0], [1], 1, 50],$$
$$(\Theta, \sigma^2) \sim NIG(\mu, \Lambda, \alpha, \beta),$$
$$\mu_x, \tau^2 \sim NIW(\mu', \Lambda', \Psi', \nu'),$$
$$X \sim N(\mu_x, \tau^2),$$
$$S := t(X, y) = [X^T X, X^T y, y^T y],$$
$$sensitivity_x = x_{max} - x_{min},$$
$$sensitivity_y = y_{max} - y_{min},$$

- privatize-suff-stats($S, sensitivity_x, sensitivity_y, \epsilon$)

$$d = S[XX].shape[0],$$
$$w_x = \Sigma(sensitivity_x[:-1])^2,$$
$$w_y = sensitivity_y{}^2,$$
$$\Delta_s = w_x{}^2 d(d+1)/2 + w_x w_y d + w_y{}^2,$$
$$z = [z_i \sim Lap(s_i, \Delta_s/\epsilon : s_i \in S)],$$

Once the noisy sufficient statistics z is computed, Section 3.2 of the paper explains the principle behind the 'Noise-naive' method:

1) Naively replace the non-private sufficient statistic s in with the noisy version z
2) Perform the conjugate update as described in Eq (1).

Because the noise-naive method is fast, we were able to set up an experiment where we compute the noise-naive posterior for N in 10, 100, 1,000, 10,000, 50,000, 10,0000, 50,0000, 1,000,000 and test if it effectively produces an asymptotically correct posterior as proposed by the authors in section 3.2. The resulting plot (Fig. 10) does confirm the authors' statement: for each parameter θ_0, θ_{bias} and σ^2, as N increases, the mean value of the posterior distribution gets closer to the non-private estimate. From $N = 500,000$ the estimates become nearly indistinguishable. The run-times figure (Fig. 11) for this experiment shows how fast the Naive method runs regardless of the sample size, which suggests that this method could be a very good candidate to consider for large enough N.

V. PERFORMANCE METRICS

Figure 3.(f) from the original paper [1] describes the method run-times for $\epsilon = 0.1$, and N in $10, 100$. For the sake of comparison, we have reproduced this plot in Fig. 12. Since we were not able to find the code that generated this figure in the GitHub repository, we implemented our custom version by improving the author's `run_methods()` function.

```
#--------------------------------------
# Pseudo-Code for the new run_methods()
#--------------------------------------
runtimes = pd.DataFrame(
    index=list(methods),
    columns=['runtime']
)
for method in methods:
    start = time.time()
    compute_posterior(method)
    end = time.time()
    runtimes.at['method','runtime'] = end-start
)
```

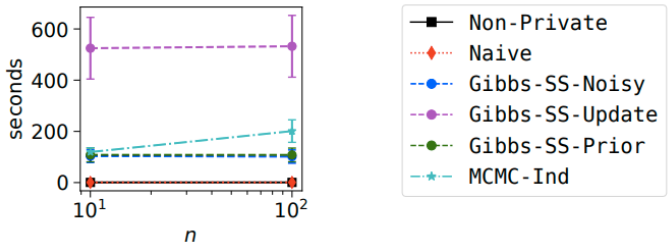

Fig. 12. Figure 3.f from the original paper

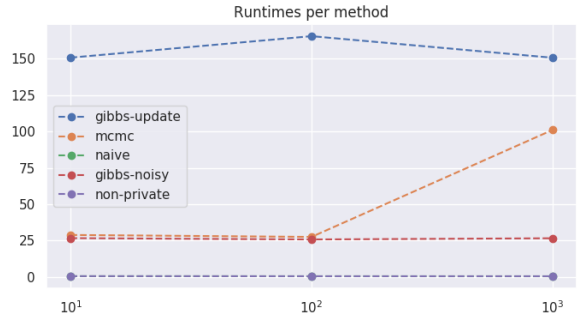

Fig. 13. Our custom reproduction of Figure 3.f from the original paper. The run-times are in seconds

In the main script (`*driver.py`) we loop through $N = 10, 100, 1000$. For each iteration, we call a custom function that plots the run-times for each method. Our version of Fig. 12 is presented in Fig. 13, where we added the case of $N = 1000$. It appears that our hardware setup is faster than the one used to run the code in the original paper, but we arrive to the same conclusion as the authors: compared to the other methods, 'Naive' and 'Non-private' are by far the fastest to run, 'Gibbs-update' is the slowest, the run-time of 'mcmc' increases proportionally to the sample size N.

VI. Conclusion and Future Research Directions

In this paper, we attempted to reproduce the results of the paper "Differentially Private Bayesian Linear Regression" by Bernstein and Sheldon [18]. Despite the fact that we were not able to replicate most of the results reported in the paper by running the code provided by the authors, we were able to recreate our own versions of some of the plots by adapting the methods described in the paper. We also performed two ablation and improvement studies. Our first experiment isolated the generation of synthetic data and posteriors and attempted to illustrate it by new plots that were not present in the published code. Our second experiment confirmed the statement from the authors according to which the Noise-naive method produces an asymptotically correct posterior.

In [18], Bernstein and Sheldon adopted the Laplace Mechanism to inject some noise into the data. This noise follows the Laplace distribution since it follows the ϵ-indistinguishability. As an extension, one can generate the noise from the generalized normal (GL) distribution, which belongs to the exponential family. In fact the Laplace and the Gaussian distribution are special cases from the GL distribution and the GL distribution satisfies the ϵ-indistinguishability [32]. Another extension for this work is to extending the methods developed for the Bayesian linear regression to the Bayesian logistic regression.

References

[1] G. Bernstein and D. R. Sheldon, "Differentially private bayesian linear regression," in *Advances in Neural Information Processing Systems*, 2019, pp. 523–533.

[2] B. Finlay and A. Agresti, *Statistical methods for the social sciences*. Dellen, 1986.

[3] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[4] J. Foulds, J. Geumlek, M. Welling, and K. Chaudhuri, "On the theory and practice of privacy-preserving bayesian data analysis," *arXiv preprint arXiv:1603.07294*, 2016.

[5] F. McSherry and I. Mironov, "Differentially private recommender systems: Building privacy into the netflix prize contenders," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 627–636.

[6] D. Vu and A. Slavkovic, "Differential privacy for clinical trial data: Preliminary evaluations," in *2009 IEEE International Conference on Data Mining Workshops*. IEEE, 2009, pp. 138–143.

[7] Y.-X. Wang, "Revisiting differentially private linear regression: optimal and adaptive prediction & estimation in unbounded domain," *arXiv preprint arXiv:1803.02596*, 2018.

[8] Z. Zhang, B. I. Rubinstein, and C. Dimitrakakis, "On the differential privacy of bayesian inference," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[9] C. Dimitrakakis, B. Nelson, A. Mitrokotsa, and B. I. Rubinstein, "Robust and private bayesian inference," in *International Conference on Algorithmic Learning Theory*. Springer, 2014, pp. 291–305.

[10] J. Geumlek, S. Song, and K. Chaudhuri, "Renyi differential privacy mechanisms for posterior sampling," in *Advances in Neural Information Processing Systems*, 2017, pp. 5289–5298.

[11] K. Minami, H. Arai, I. Sato, and H. Nakagawa, "Differential privacy without sensitivity," in *Advances in Neural Information Processing Systems*, 2016, pp. 956–964.

[12] Y.-X. Wang, S. Fienberg, and A. Smola, "Privacy for free: Posterior sampling and stochastic gradient monte carlo," in *International Conference on Machine Learning*, 2015, pp. 2493–2502.

[13] C. Dwork and A. Smith, "Differential privacy for statistics: What we know and what we want to learn," *Journal of Privacy and Confidentiality*, vol. 1, no. 2, 2010.

[14] A. Smith, "Efficient, differentially private point estimators," *arXiv preprint arXiv:0809.4794*, 2008.

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

[16] 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*. IEEE, 2014, pp. 464–473.

[17] O. Sheffet, "Differentially private ordinary least squares," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3105–3114.

[18] A. F. Barrientos, J. P. Reiter, A. Machanavajjhala, and Y. Chen, "Differentially private significance tests for regression coefficients," *Journal of Computational and Graphical Statistics*, pp. 1–24, 2019.

[19] A. Honkela, M. Das, A. Nieminen, O. Dikmen, and S. Kaski, "Efficient differentially private learning improves drug sensitivity prediction," *Biology direct*, vol. 13, no. 1, p. 1, 2018.

[20] J. Jälkö, O. Dikmen, and A. Honkela, "Differentially private variational inference for non-conjugate models," *arXiv preprint arXiv:1610.08749*, 2016.

[21] M. Park, J. Foulds, K. Chaudhuri, and M. Welling, "Variational bayes in private settings (vips)," *arXiv preprint arXiv:1611.00340*, 2016.

[22] G. Bernstein and D. R. Sheldon, "Differentially private bayesian inference for exponential families," in *Advances in Neural Information Processing Systems*, 2018, pp. 2919–2929.

[23] ——, "private bayesian regression," [Online]. Available: https://github.com/gbernstein6/private_bayesian_regression, 2019.

[24] A. C. Rencher and M. Schimek, "Methods of multivariate analysis," *Computational Statistics*, vol. 12, no. 4, pp. 422–422, 1997.

[25] M. G. Kendall, A. Stuart, J. K. Ord, S. F. Arnold, A. O'Hagan, and J. Forster, *Kendall's advanced theory of statistics*. Griffin London, 1987, vol. 1.
[26] G. Casella and R. L. Berger, *Statistical inference*. Duxbury Pacific Grove, CA, 2002, vol. 2.
[27] J. Awan and A. Slavkovic, "Structure and sensitivity in differential privacy: Comparing k-norm mechanisms," *arXiv preprint arXiv:1801.09236*, 2018.
[28] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett, "Functional mechanism: regression analysis under differential privacy," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1364–1375, 2012.
[29] P. J. Bickel and K. A. Doksum, *Mathematical Statistics: Basic Ideas and Selected Topics, Volumes I-II Package*. Chapman and Hall/CRC, 2015.
[30] T. Park and G. Casella, "The bayesian lasso," *Journal of the American Statistical Association*, vol. 103, no. 482, pp. 681–686, 2008.
[31] F. J. Massey Jr, "The kolmogorov-smirnov test for goodness of fit," *Journal of the American statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.
[32] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," *Journal of Privacy and Confidentiality*, vol. 7, no. 3, pp. 17–51, 2016.