MONTE CARLO METHODS FOR DATA PRIVACY APPLICATIONS

by
BARIŞ ALPARSLAN

**Submitted to the Graduate School of Engineering and
Natural Sciences in partial fulfilment of
the requirements for the degree of Master of Science**

**Sabancı University
May 2023**

ABSTRACT

MONTE CARLO METHODS FOR DATA PRIVACY APPLICATIONS

BARIŞ ALPARSLAN

DATA SCIENCE M.A. THESIS, MAY 2023

Thesis Supervisor: Asst. Prof. Sinan Yıldırım

Keywords: Bayesian inference, Differential privacy, MCMC

This thesis focuses on data privacy applications with Bayesian inference, particularly Markov chain Monte Carlo (MCMC) methods for two main data privacy problems. Firstly, we focus on statistic selection with a Fisher information, and we show that informativeness and efficiency are closely related in the differential privacy setting. Then, we propose a novel generative model for the private linear regression that outshines state-of-art methods.

In this work, MCMC algorithms are specifically developed for several data privacy settings. While some of the settings enable to work with simple and efficient Metropolis-Hastings (MH), others require more advanced sampling methods such as Pseudo-Marginal Metropolis-Hastings (PMMH), Metropolis-Hastings with Averaged Acceptance Ratios (MHAAR) or MH-within-Gibbs sampling. In detail, we prefer using versions of MH, PMMH, MHAAR for the statistic selection, and derivatives of the MH-within-Gibbs for the linear regression problem.

At the end, we conduct several numerical experiments for evaluation purposes. In the statistic selection part, we rigorously deal with each problem setting and we obtain that Fisher information is actually a useful tool for the differential privacy applications for almost all possible problem definitions. For the linear regression, both simulated and real datasets are tested, and we observe that proposed methods beat existing algorithms in terms of efficiency and effectiveness.

ÖZET

MONTE CARLO METODLARIYLA VERI MAHREMIYETI UYGULAMALARI

TEZ YAZARI

VERİ BİLİMİ YÜKSEK LİSANS TEZİ, MAYIS 2023

Tez Danışmanı: Dr.Öğr.Üyesi Sinan Yıldırım

Anahtar Kelimeler: Bayesci çıkarım, Diferansiyel mahremiyet, MCMC

Bu tez, iki ana veri mahremiyeti problemine Bayesci çıkarım yöntemlerini kullanarak odaklanmaktadır. Bayesci çıkarım yöntemleri arasından özellikle Markov chain Monte Carlo (MCMC) ve bu metodu temel alan yaklaşımlar incelenmektedir. Bahsedilen problemlerden ilki, veri mahremiyeti uygulamalarında kullanıcıya sunulacak olan gizli verinin seçilmesi ile ilgilidir ve bunun için Fisher information yöntemi önerilmektedir. Bu noktada bilgilendiricilik ile çıkarım metodlarının başarısının yakından ilişkili olduğunu gösterilmiştir. Ardından, veri mahremiyetini gözeten doğrusal regresyon için bilinen yöntemlerden daha başarılı olan yeni bir üretici model önerilmiştir.

Bu çalışmada, MCMC algoritmaları, çeşitli veri mahremiyeti senaryoları için özel olarak geliştirilmiştir. Bazı senaryolar basit ve verimli Metropolis-Hastings (MH) ile çalışmayı mümkün kılarken, diğerleri Pseudo-Marginal Metropolis-Hastings (PMMH), Metropolis-Hastings with Averaged Acceptance Ratios (MHAAR) veya Gibbs örnekleme gibi daha gelişmiş örnekleme yöntemleri gerektirmektedir. Daha ayrıntılı olarak, istatistik seçimi için MH, PMMH, MHAAR gibi algoritmalar uygulanırken, doğrusal regresyon problemi için Gibbs örnekleme ve türevlerinin kullanılması tercih edilmiştir.

Sonunda, farklı durumları kapsayan sayısal deneyler gerçekleştirilmiştir. İstatistik seçimi bölümünde, her bir senaryo üzerinde titizlikle durulmuş ve Fisher information yönteminin hemen hemen tüm olası problem tanımlarında diferansiyel mahremiyet uygulamaları için faydalı bir araç olduğu gösterilmiştir. Doğrusal regresyon için hem simüle edilmiş hem de gerçek veri kümeleri kullanılmış ve önerilen yöntemlerin verimlilik ve etkinlik açısından mevcut algoritmaları geride bıraktığı gözlemlenmiştir.

ACKNOWLEDGEMENTS

This work would not have been possible without my advisor, Prof. Sinan Yıldırım. I would like to thank him and express my appreciation for his patient guidance and valuable support throughout the journey. His wisdom and endurance taught me a lot. I am sure that it will always help me in my future endeavours.

Also, words cannot express my gratitude to Prof. İlker Birbil for his invaluable advices and contributions. Without his guidance, I would not have found motivation to continue my academic studies.

Special thanks should also go to Prof. Yücel Saygın for his participation to the thesis defense committee and his priceless feedbacks.

Finally, I am always grateful to my family for their life-time support. They have always encouraged me regardless of my choices and I can not express how precious this is. I have learnt a lot from their experiences and will continue to do so.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. Introduction

Leveraging vast amount of data has transformed and boosted many operations in a world of technological developments. However, rapid digitalization comes with an undesirable result of violating human rights and the modern technology has repeatedly failed to protect sensitive personal information (Human, 2022; Isaak & Hanna, 2018; Matte, Bielova & Santos, 2020; Trautman, 2022). Hence, it is utterly important for the researchers and the practitioners to focus on developing privacy-preserving data analytics solutions for the sustainable and efficient future technologies on data science and machine learning. In fact, along with the discussion in this thesis, various data privacy techniques have been suggested for preserving sensitive information (Darwish, Essa, Osman & Ismail, 2022).

Among many privacy-preserving approaches, one of them, differential privacy, outshines other methods as it effectively enables exploiting sensitive data without violating the personal information with certain mathematical guarantees (Dwork, Roth & others, 2014). Therefore, this thesis particularly focuses on differential privacy applications while inferring valuable information from sensitive data to contribute the research on data privacy technologies.

Differential privacy has also edge over other privacy techniques as many researchers showed that it is adaptive and can be safely implemented to the well-known data analytics and machine learning methods ranging from fundamentals such as regression or classification to the advanced neural models such as generators or large language models with transformers (Zhao & Chen, 2022a). In this regard, this thesis discusses Bayesian inference methods, especially Markov chain Monte Carlo (MCMC), for the differential privacy applications as they have proven to be promising by several researchers (Heikkilä, Jälkö, Dikmen & Honkela, 2019; Yıldırım & Ermiş, 2019).

1.1 Contribution of the thesis

As it was mentioned above, this thesis particularly aims to combine differential privacy with the Bayesian inference. In this context, researchers have proposed several methods for differentially private Bayesian estimation (Bernstein & Sheldon, 2019; Heikkilä et al., 2019; Räisä, Koskela & Honkela, 2021; Wang, 2018; Wang, Fienberg & Smola, 2015). However, this thesis come up with novel approaches and extend the literature. Specifically, whereas none of the works in the literature has focused on the informativeness of the statistics for the differentially private analyis, a part of this thesis exactly closes this gap by utilizing Fisher information for selecting the best statistic for the differentially private inference. Indeed, we show that Fisher information works unexpectedly well to this end in the following sections. Statistic selection methodology is especially important for practical purposes. Using simple distributions, we show that the conventional statistics may not be the best choices to share the data privately.

In addition to the statistic selection technique, this thesis also proposes a novel Bayesian approach for one of the oldest problems in the privacy literature, privacy-preserving linear regression. A new generative hierarchical model with unique distributional relations forms the foundation of the proposed method for the private linear regression setting. The model simply corrupts summary statistics and samples from the posterior distribution given those perturbated summary statistics using MCMC technique. For brevity, the contributions of this thesis can be summarized as:

- An unique statistic selection methodology using Fisher information. In detail, we rigorously show that the most informative statistic in terms of Fisher information results in better performance when it is combined with Bayesian inference with differential privacy.
- An efficient sampling method based on a novel hierarchical structure for the private linear regression problem. In short, newly developed algorithms both enable to work on distributed data environment, which is crucially important in a digitalized era, and satisfactorily beat existing methods.

1.2 Outline

There are mainly four chapters in this thesis. While the first two chapters present the foundations of this study on Markov chain Monte Carlo and differential privacy, the following chapters discuss the proposed methods in detail and provide

well-rounded numerical experiments for the justifications. In detail, chapter 2 is all about the definitions and the literature about Bayesian inference and Markov chain Monte Carlo where one can find the rigorous rigorous explanations about the utilized models. Then, chapter 3 provides details of differential privacy in depth so that one can easily grasp the idea of differential privacy with sufficiently provided definitions. Given the basics, chapter 4 is the first chapter presenting one of the contributions of this thesis, the statistic selection methodology. In this part, various privacy settings are considered to come up with a viable statistic selection method based on Fisher information. On top of the definitions, chapter 5 discusses possible Bayesian inference methods complementing the statistic selection methodology and the linear regression problem. Finally, chapters 6 and 7 are reserved for the numerical experiments.

2. Markov chain Monte Carlo

Bayesian inference usually requires employing complex probability distributions (Brooks, 1998). One may need to clearly describe or integrate over those complex distributions for obtaining the marginal distribution/expectation in order to effectively utilize those distributions in the inference. Due to the intractability and complexity of these integrals in most of the cases, numerical approximations are essential (Brooks, 1998; Smith, 1991). Here, one useful Bayesian inference method is Markov chain Monte Carlo (MCMC).

The idea of MCMC is based on constructing an elegant Markov chain that has invariant distribution as the target posterior distribution. With sufficiently large run, samples coming from invariant distribution of the Markov chain can be treated as instances of target distribution (Brooks, 1998). Those samples from target distribution can then be used as numerical approximations of the required integrals or expectations.

At this point, we need to go further and discuss Markov chains as they constitute the foundations of the main methodology of this thesis. Markov chains can be considered as a sequence of random variables that has the Markov property, which means that the next sample from the chain is not affected by the past samples except the current one. More formal definition is

Definition 1 (Markov chain (Brooks, Gelman, Jones & Meng, 2011)) *Consider a sequence of random variables in set* $S = \{X_1, X_2 \ldots\}$. *S forms a Markov chain if for all n*

$$P(X_{n+1}|X_n, \ldots, X_1) = P(X_{n+1}|X_n).$$

Markov chains have unique limiting distributions under some conditions, and these conditions are especially important while designing a MCMC algorithm to ensure true convergence. Firstly, the chain should be irreducible, which means that any state n can be reached by any other state for all n in a finite number of steps. Secondly, the chain should be aperiodic, which means the common divisor of required

steps to return back to the same state is 1. Finally, it should be positive recurrent, i.e. the number of steps taken for transition back to a same state is expected to be finite for the existence of stationary distribution (Spade, 2020). The stationary distributions are important as one can possibly design a chain that has specific distribution of π as the limiting distribution, and use samples from that limiting distribution to explore the target distribution in the inference scheme.

In addition to the specifications above, most of the MCMC algorithms are designed to satisfy reversibility and detailed balance as it ensures the existence of the desired limiting distribution (Sharma, 2017). Detailed balance for a transition kernel Q with respect to a distribution π is satisfied when stationary distribution π is

$$\pi(y)Q(x|y) = \pi(x)Q(y|x) \quad \forall x, y \in \mathcal{X}. \tag{2.1}$$

There are several well-known techniques to design such Markov chains whose limiting distribution is target distribution $\pi(\cdot)$.

2.1 Metropolis-Hastings

Metropolis-Hastings is one of the most fundamental and well-known methods for the purpose of designing a Markov chain with a desired limiting distribution of $\pi(.)$. Roughly speaking, it requires proposal distribution as an input (Flötteröd & Bierlaire, 2013; Hastings, 1970) to propose a sample from this distribution in each iteration. Then, with a specific acceptance probability these proposals are either considered as a part of the desired target distribution or they are disregarded.

One iteration of a typical MH algorithm consists of three steps and these steps are repeated until a specified iteration number is exceeded.

1.1 Propose a new sample θ' using current step θ from proposal distribution $q(.)$.

1.2 Calculate acceptance probability $\alpha(\theta, \theta') = \min\{1, \frac{\pi(\theta')q(\theta|\theta')}{\pi(\theta)q(\theta'|\theta)}\}$.

1.3 Accept the proposal with probability $\alpha(\theta, \theta')$; otherwise, reject the proposal. When the proposal is accepted, the current state of the Markov chain is set to θ', otherwise the current state is not updated and stays θ.

Note that, the acceptance probability in step 2 comes from the detailed balance in

equation (2.1). In detail, consider a transition probability from state x^t to $x^{t+1} \neq x^t$ as $Q(x^{t+1}|x^t)$, and $\pi(x)$ as the limiting distribution of the Markov chain. By the reversibility, we have

$$Q(x^{t+1}|x^t)\pi(x^t) = Q(x^t|x^{t+1})\pi(x^{t+1}).$$

Then, we can also write transition probability $Q(x^{t+1}|x^t)$ as

$$Q(x^{t+1}|x^t) = q(x^{t+1}|x^t)\alpha(x^{t+1}, x^t),$$

where $q(.)$ is proposal distribution given the current state, and $\alpha(x^{t+1}, x^t)$ is the probability of accepting x^{t+1}. Then, we can derive that

$$\begin{aligned} \frac{Q(x^{t+1}|x^t)}{Q(x^t|x^{t+1})} &= \frac{\pi(x^{t+1})}{\pi(x^t)}, \\ \frac{q(x^{t+1}|x^t)\alpha(x^{t+1}, x^t)}{q(x^t|x^{t+1})\alpha(x^t, x^{t+1})} &= \frac{\pi(x^{t+1})}{\pi(x^t)}, \\ \frac{\alpha(x^{t+1}, x^t)}{\alpha(x^t, x^{t+1})} &= \frac{q(x^t|x^{t+1})\pi(x^{t+1})}{q(x^{t+1}|x^t)\pi(x^t)}. \end{aligned} \tag{2.2}$$

Then, using idea from Hastings (1970); Metropolis, Rosenbluth, Rosenbluth, Teller & Teller (1953) and equation (2.2), one can accept the proposal with $\alpha(\theta, \theta')$ as in step 2. As a side note, it is possible to use a different version of acceptance probability by still satisfying detailed balance (Barker, 1965), but the $\alpha(\theta, \theta')$ in step 2 is proven to be optimal by resulting in less rejection of convenient proposals (Brooks, 1998; Peskun, 1973). Additionally, the proposal mechanism in step 1 is a design parameter, and it may affect the efficiency of the mechanism. Some of well-known proposal mechanisms that can be utilized effectively are (Sharma, 2017) given below, and Algorithm 1 summarizes one iteration of Metropolis-Hastings algorithm.

- **Symmetric proposals:** $q(\theta'|\theta) = q(\theta|\theta')$, and $\alpha(\theta, \theta') = \min\{1, \frac{\pi(\theta')}{\pi(\theta)}\}$.
- **Random walk**: Propose $\theta' = \theta + \epsilon$, where ϵ is a probability density usually taken as normal or uniform (Chains, 2010). When it is normally distributed, proposal distribution is $\theta'|\theta \sim \mathcal{N}(\theta, \sigma_q^2)$.
- **Independence sampler:** $q(\theta'|\theta) = q(\theta')$.

2.2 Pseudo-Marginal Metropolis-Hastings

Algorithm 1 Metropolis-Hastings algorithm

Begin with some $\theta^{(0)}$
for $i = 1, 2, \ldots$ **do**
  Propose $\theta' \sim q(\theta'|\theta^{(i-1)})$
  Accept θ' and return $\theta^{(i)} = \theta'$ with probability of $\alpha(\theta^{(i-1)}, \theta')$
  else reject the proposed value and return $\theta^{(i)} = \theta^{(i-1)}$

Pseudo-Marginal Metropolis-Hastings (PMMH) is another method aiming to sample from a target distribution using the limiting distribution of a Markov chain. However, PMMH further extends the MH algorithm by employing importance sampling (Andrieu & Roberts, 2009), and enables to make inference with likelihood-free approach (Warne, Baker & Simpson, 2020). In a way, PMMH mimics the original MH algorithm (Deligiannidis, Doucet & Pitt, 2018) with the point-wise estimation of likelihoods instead of the true likelihood, and it is more applicable to the real cases where tractable likelihood functions are not usually available (Warne et al., 2020). As a side note, it is sometimes called as exact-approximate MCMC method as they are known to converge exact posterior distribution with an approximation of ideal (but intractable) MH algorithm.

Importance sampling enables estimating the mathematical expectation of target distribution using weighted averages of random variables from another distribution (Tokdar & Kass, 2010). Suppose that target distribution is $\pi(\theta)$, for some $f(\theta)$ we know that the following equation holds true by Monte Carlo integration (Brooks et al., 2011);

$$\mathbb{E}_\pi[f(\theta)] = \int f(\theta)\pi(\theta)d\theta \approx \frac{1}{N}\sum_{i=1}^{N} f(\theta^{(i)}), \quad \theta^{(i)} \sim \pi(\theta). \tag{2.3}$$

For some well-defined $q(.)$ distribution that is satisfying $q(\theta) > 0$ when $\pi(\theta) > 0$, we can write the integration in equation (2.3) as

$$\mathbb{E}_\pi[f(\theta)] = \int f(\theta)q(\theta)\frac{\pi(\theta)}{q(\theta)}d\theta.$$

Realize that, this integration is same with taking expectation of $f(\theta)\frac{\pi(\theta)}{q(\theta)}$ according to the $q(.)$, which is

$$\int f(\theta)\frac{\pi(\theta)}{q(\theta)}q(\theta)d\theta = \mathbb{E}_q[f(\theta)w(\theta)], \text{ where } w(\theta) = \frac{\pi(\theta)}{q(\theta)}.$$

Then, by using the same Monte Carlo integration strategy, we can approximate this

integration as

$$\mathbb{E}_q[f(\theta)w(\theta)] \approx \frac{1}{N}\sum_{i=1}^{N} f(\theta^{(i)})w(\theta^{(i)}), \quad \theta^{(i)} \sim q(\theta). \tag{2.4}$$

Therefore, it is possible to estimate $E_\pi[f(\theta)]$ using samples from $q(.)$ as in equation (2.4). The most important advantage of importance sampling is that sampling from $q(.)$ is a lot easier than sampling from $\pi(.)$ as it is a well-defined and tractable distribution compared to $\pi(.)$ by design. On the other hand, in most of the real-life cases, the target distribution $\pi(\theta)$ and/or $q(\theta)$ are known up to some normalizing constant where drawing true samples may not be possible. At this point, Hesterberg (1995) proposed a more capable version of the importance sampling called as self-normalized importance sampling. In this case, one can make small adjustment on the importance sampling scheme. Assume that we have $\pi(\theta) = \lambda_0\pi_0(\theta)$ where only $\pi_0(\theta)$ is known. Using the self-normalized importance methodology it is possible to change weights $w(\theta)$ with $w_0(\theta) = \frac{\pi_0(\theta)}{q_0(\theta)}$. Then, the final Monte Carlo estimation step becomes

$$\frac{\sum_{i=1}^{N} f(\theta^{(i)})w_0(\theta^{(i)})}{\sum_{i=1}^{N} w_0(\theta)}, \quad \theta^{(i)} \sim q(\theta). \tag{2.5}$$

Coming back to the discussion of PMMH, suppose we have a likelihood function $p(y|\theta)$, and suppose we need to use auxilary variable x for the sampling procedure, then we have (Drovandi, Moores & Boys, 2018)

$$p(y|\theta) = \int p(y|x,\theta)p(x|\theta)dx = \mathbb{E}_{x|\theta}[p(y|x,\theta)] \approx \frac{1}{N}\sum_{i=1}^{N} p(y|x_i,\theta), \;\; x^{(i)} \sim p(x|\theta).$$

Combining equations (2.4) and (2.5), it is possible to replace this likelihood with an unbiased estimator using self-normalized importance sampling as (Andrieu & Roberts, 2009; Beaumont, 2003; Drovandi et al., 2018)

$$\hat{p}(y|\theta) = \frac{1}{N}\sum_{i=1}^{N} \frac{p(y|x_i,\theta)p(x_i|\theta)}{q(x_i)}, \quad x_i \sim q(x), \tag{2.6}$$

which can be replaced with the true-likelihood in the Algorithm 1.

All in all, a generic version of PMMH algorithm is shown in Algorithm 2 where target distribution is $\hat{\pi}(\theta) \propto \hat{p}(y|\theta)p(\theta)$.

2.3 Metropolis-Hastings with Averaged Acceptance Ratio

Algorithm 2 Pseudo-marginal Metropolis-Hastings algorithm

Begin with some $\theta^{(0)}, \hat{Z}^{(0)}$

for $i = 1, 2, \ldots$ **do**

Propose $\theta' \sim q(.|\theta^{(i-1)})$

Propose $x^j \sim q_{\theta'}(.)$ for j = 1, 2,..., J

Calculate $\hat{Z}' = \frac{1}{N}\sum_{j=1}^{N} \frac{p(y|x^{(j)},\theta^{(i-1)})p(x^{(j)}|\theta^{(i-1)})}{q_{\theta'}(x^{(j)})}$

Accept θ' and return $\theta^{(i)} = \theta', \hat{Z}^{(i)} = \hat{Z}'$ with probability

$$\min\{1, \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{p(\theta')}{p(\theta)} \frac{\hat{Z}'}{\hat{Z}^{(i-1)}}\}$$

else reject the proposed variables and return $\theta^{(i)} = \theta^{(i-1)}, \hat{Z}^{(i)} = \hat{Z}^{(i-1)}$

Although PMMH extends the capability of the MH algorithm by drawing samples with estimated likelihood function, it may be computationally inefficient when the data size grows since the number of iterations required to converge desired distribution is also skyrocketed relatedly (Deligiannidis et al., 2018). Additionally, the PMMH algorithm carries Z from the previous iteration, which may result in rejections for consecutive iterations, so its Markov chain eventually may become sticky.

One way to overcome the problem of the sticky chain is designing an MCMC algorithm with correlated random variables and plugging them in the marginal acceptance ratio of the algorithm. The idea behind using correlated auxiliary random variables in the likelihood function is that mimicking the original MH algorithm with a less variated acceptance ratio (Deligiannidis et al., 2018). With a less variated acceptance ratio, this algorithm partially mitigates the problem of stickiness in PMMH. Algorithm 3 demonstrates one iteration of the correlated pseudo-marginal (CPM) method from Deligiannidis et al. (2018).

Realize that CPM still carries U from the previous iteration if the proposal is not accepted as in algorithm 3. Hence, the problem of stickiness is not completely resolved. Here, a more recent class of exact-approximate MCMC algorithm called MH with Averaged Acceptance Ratios (MHAAR) takes the stage (Andrieu, Yıldırım, Doucet & Chopin, 2020). MHAAR also employs the likelihood-free approach like PMMH and CPM, but it (almost) fully updates both numerator and the denominator of the acceptance ratio independent from the decision.

There are multiple versions of MHAAR, but in the following parts of the thesis, we particularly mention the algorithm MHAAR-RB, which employs a "Rao-Blackwellised" acceptance ratio (Andrieu et al., 2020). To elaborate, the Rao-Blackwell theorem states that the expected value of the conditional distribution

Algorithm 3 Correlated Pseudo-Marginal (CPM) - one iteration (Deligiannidis et al., 2018)

Begin with some ρ, θ, U

for $i = 1, 2, \ldots$ **do**

Propose $\theta' \sim q(.|\theta)$

Sample $\varepsilon \sim \mathcal{N}(0_d, I_d)$, and $U' = \rho U + \sqrt{1-\rho^2}\varepsilon$

Calculate $\hat{p}(y|\theta', U')$, which is an estimation of $p(y|\theta')$

Accept (θ', U') and return $\theta = \theta', U = U'$ with probability

$$\min\{1, \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{p(\theta')}{p(\theta)} \frac{\hat{p}(y|\theta', U')}{\hat{p}(y|\theta, U)}\}$$

else reject the proposed values and return the previous ones as $\theta = \theta, U = U$

$\mathbb{E}[t|u]$ is a less variant estimator of θ where t is an unbiased estimation and u is the sufficient statistics of the target parameter θ, and this idea is especially effective in particle filtering sequential Monte Carlo methodology (Blackwell, 1947; Robert & Roberts, 2021). "Rao-Blackwellised" acceptance ratio is (Andrieu et al., 2020);

$$\alpha(\theta', \theta) = \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{\eta(\theta')}{\eta(\theta)} \frac{\sum_j w_j}{\sum_j w'_j}, \tag{2.7}$$

where $w_j = L(y, \theta, t^{(j)})/q_{\theta,\theta'}(t^{(j)})$, and $w'_j = L(y, \theta', t^{(j)})/q_{\theta,\theta'}(t^{(j)})$, and L is a likelihood function regarding the parameter and the data. All in all, MHAAR algorithm is demonstrated in algorithm 4. Note that, correctness of the MHAAR methodology is proven in Andrieu et al. (2020).

Algorithm 4 MH with Averaged Acceptance Ratio (MHAAR) - one iteration (Andrieu et al., 2020)

N is sample size for u, y is data, and begin with some (θ, t)

Propose $\theta' \sim q(.|\theta)$

for $j = 1, 2, \ldots, N$ **do**

If $j = 1$ set $t^{(1)} = t$, or sample $t^{(j)} \sim q_{\theta',\theta}(.)$

Calculate $w_j = \frac{L(y,\theta,t^{(j)})}{q_{\theta,\theta'}(t^{(j)})}$, and $w'_j = \frac{L(y,\theta',t^{(j)})}{q_{\theta,\theta'}(t^{(j)})}$

Find $\min\{1, \alpha(\theta', \theta)\}$ using equation (2.7)

With this probability sample $k \in \{1, \ldots, N\}$ with probability proportional to w'_k and return $(\theta', t^{(k)})$.

Else reject and sample $k \in \{1, \ldots, N\}$ with probability proportional to w_k, and return $(\theta, t^{(k)})$.

2.4 Gibbs Sampling

Among numerous MCMC algorithms, another well-known class of MCMC algorithms is Gibbs sampling, stemming from the idea of sampling with tractable conditional distributions rather than focusing on complicated joint distributions. This idea was first exploited in a formal manner in 1984, and it was proven that this method converges to the desired posterior distribution just like MH algorithm (Casella & George, 1992; Gelfand, 2000; Geman & Geman, 1984). One can classify Gibbs sampling as a special case of MH methodology where the proposals are always accepted, i.e. no accept-reject mechanism, but the concern is designing a Markov chain whose invariant distribution converges to the target posterior distribution using full-conditional distributions (probability of a random variable conditioned on all other random variables) (Walsh, 2004).

To elaborate, suppose we have a vector of random variables as $(\theta_1, \ldots, \theta_n)$, and t corresponds to the step of the algorithm as $\theta^{(t)} = (\theta_1^{(t)}, \ldots, \theta_n^{(t)})$. Additionally, consider the full-conditional distributions as (Gelfand, 2000)

$$\pi\left(\theta_i | \theta_j \; ; \forall j \in \{1, \ldots, N\} \text{ where } i \neq j\right) \quad \forall i \in \{1, \ldots, N\}.$$

Then, in each iteration, the algorithm samples $\theta_i^{(t)}$ from the conditional distribution

$$\pi\left(\theta_i^{(t)} | \theta_j^{(t)}, \theta_k^{(t-1)} \; ; \forall j < i, \forall k > i\right).$$

Usually, employing conjugate priors enables tractable full-conditional distributions which are easier to sample from while there are other ways of drawing samples from the full-conditionals when they are not known analytically Gelfand (2000). All in all, algorithm 5 demonstrates the Gibbs sampling methodology clearly.

Algorithm 5 Gibbs sampler

begin with some $\theta^{(0)} = (\theta_1^{(0)}, \ldots, \theta_n^{(0)})$
for $t = 1, \ldots, T$ **do**
  for $j = 1, 2, \ldots, n$ **do**
    Sample $\theta_j^{(t)}$ from $\pi_j\left(\theta_j^{(t)} | \theta_m^{(t)}, \theta_k^{(t-1)} \; ; \forall m < j, \forall k > j\right)$

It is possible to show that Gibbs sampler converges to an invariant distribution π. For this purpose, assume $\theta = (\theta_1, \ldots, \theta_n), and \; \tilde{\theta} = (\theta_m, \theta_k \; ; \forall m < j, \forall k > j)$, i.e. all other parts of θ other than θ_j. Consider, T_j represents transition for each step with

$j = 1, \ldots, n$. Then, $T = T_1 T_2 \ldots T_n$ is a transition kernel for the Gibbs sampler as it applies each T_j sequentially. With a T_j that satisfies the detailed balance in equation (2.1), we have $\pi T_j = \pi$. Therefore, transition kernel T can be written as

$$\pi T = \pi T_1 T_2 \ldots T_n = (\pi T_1) T_2 \ldots T_n = \pi T_2 \ldots T_n = \ldots = \pi.$$

Hence, π is an invariant distribution for Gibbs sampler with transition kernel of T that satisfies the detailed balance.

2.5 Improvements on Gibbs sampling

2.5.1 Collapsed Gibbs sampling

There are some concerns about the Gibbs sampling that has been an active field of research. One of these drawbacks is the correlation among the samples of the Gibbs updates as it results in poor and slow convergence to the desired distribution. One way to overcome this problem is collapsing (integrating-out) one or two components from the chain Liu, Wong & Kong (1994). One may simply consider collapsing the general Gibbs scheme as removing the not particulary interested variables from the sampling chain Liu et al. (1994); Park & Lee (2022a), which can be shown as

$$\begin{aligned} &\text{Sample from:} \quad \pi(x|y,z),\ \pi(y|x,z),\ \pi(z|x,y) \quad \text{(General)} \\ &\text{Sample from:} \quad \pi(x|y),\ \pi(y|x) \qquad\qquad\qquad\quad \text{(Collapsed)} \end{aligned}$$

The collapsed chain targets sampling from the marginalized version of complete joint distribution by maintaining the functional compatibility Park & Lee (2022b).

2.5.2 MH-within-Gibbs Sampling

Another problem is related to the key component of algorithm: full-conditional distributions. Although Gibbs sampler is quite powerful tool for drawing samples from

complex joint distributions, existence of the tractable conditional distributions is crucially important for the sampling methodology. Indeed, easy-to-sample conditional distributions may not be possible to obtain in a real-life scenario. In this case, one step MH can be placed inside of a Gibbs sampling Gilks & Spiegelhalter (1996); Martino, Read & Luengo (2015); Millar & Meyer (2000), and this is called MH-within-Gibbs sampling. In other words, instead of directly sampling θ_j from $\pi(\theta_j|\tilde{\theta})$, replacing it with a single step MH move whose invariant distribution is target posterior distribution $\pi(\theta_j|\tilde{\theta})$ is still a valid sampling method. Although initial samples from Markov chain don't necessarily belong to the limiting distribution, they converge to the desired posterior distribution in the later stages of outer Gibbs steps, so there is no need for extra iterations for the inner MH step.

3. Differential privacy

Differential privacy (DP) is a novel and increasingly popular privacy protection paradigm in the recent years (Dankar & Emam, 2013), and main motivation is protecting the sensitive information while enabling effective learning about the population (Dwork et al., 2014). Differential privacy definition is based on a probabilistic difference between the outputs of a randomised algorithm with inputs of two neighboring datasets. What we mean by neighboring dataset is that Hamming distance between those datasets is equal to 1. More formally,

Definition 2 (Hamming distance (Dwork et al., 2014; Zhao & Chen, 2022b)) *Hamming distance measures the difference between two datasets by counting number of rows in which they differ as*

$$\|x - x'\|_H = \sum_{i=1}^{N} \mathbb{1}_{x_1 \neq x_2}$$

where $x, x' \in \mathscr{X} = \cup_{n=1}^{\infty} \mathcal{X}^n$*, and* $x = (x_1, \ldots, x_n) \in \mathcal{X}^n$*.*

Definition 3 (ϵ-Differential Privacy (Dwork, 2008; Dwork et al., 2014)) *A randomized algorithm* M *satisfies* ϵ*-differential privacy if for all datasets* $x, x' \in \mathscr{X}$*, which satisfy* $\|x - x'\|_H = 1$*, and* $S \subseteq Range(M)$

$$Pr[M(x) \in S] \leq e^{\epsilon} \times Pr[M(x') \in S]$$

Also, a more relaxed version of this definition is available (Dankar & Emam, 2013). Instead of strictly forcing the ratio to be less than e^{ϵ}, it is possible to allow some violations on the previous definition with bound δ as in definition 4.

Definition 4 (ϵ, δ-Differential Privacy (Dwork et al., 2014)) *A randomized algorithm* M *satisfies* (ϵ, δ)*-differential privacy if for all datasets* $x, x' \in \mathscr{X}$*, which are differing at most one element, and* $S \subseteq Range(M)$

$$Pr[M(x) \in S] \leq e^{\epsilon} \times Pr[M(x') \in S] + \delta$$

It is worth noting that differential privacy is strong in the sense of randomization, but not an "absolute guarantee" about the privacy of the individuals. Instead, it injects well-designed noise in the process of inference so that the data maintain usability while the noise limits the leakage of the true sensitive information (Dwork, 2008). Another important point is that one can easily calibrate the level of privacy by changing the value of ϵ or δ. For instance, smaller ϵ means that privacy concern is high and it is quite undesirable to leak any information. Similarly, a smaller value of δ also implies a similar constraint on privacy level.

Most of the time private data analysis with differential privacy use output perturbation mechanisms, i.e. adding some noise on top of the output of a certain randomised algorithm (Nissim, Raskhodnikova & Smith, 2007), and some of the key definitions for differential privacy are presented in the following sections.

3.1 Sensitivity

Sensitivity indicates how large the noise should be for the sake of satisfying privacy constraints Dwork et al. (2014).

Definition 5 (l-p Sensitivity (Dwork, McSherry, Nissim & Smith, 2006)) *For a function $f : \mathscr{X} \to R^d$, $\Delta_{f,p}$ of a function f is defined as*

$$\Delta_{f,p} = \sup_{x_{1:n}, x'_{1:n} : \|x_{1:n} - x'_{1:n}\|_H = 1} \|f(x_{1:n}) - f(x'_{1:n})\|_p,$$

where $x, x' \in \mathscr{X}$.

Note that, the sensitivity in definition 5 is called "Global sensitivity" Nissim et al. (2007), and it is also possible to define the global sensitivity using Euclidian norm as well (Avella-Medina, 2021), so called l-2 sensitivity. Although global sensitivity is at the core of most of the differential privacy discussions, for some data queries, global sensitivity may be inefficient as it is likely to introduce undesirably large noise due to independence from the observed values (Nissim et al., 2007; Sun, Zhou, Yu & Xiong, 2020). At this point, the local sensitivity takes the stage by exploiting the observed values in the dataset. More formal expression is in definition 6.

Definition 6 (Local sensitivity (Nissim et al., 2007)) *For $f : \mathscr{X} \to R^d$,*

$\Delta_{f,local}$ *at* x *is the smallest value that satisfies*

$$\|f(x) - f(x')\|_1 \leq \Delta_{f,local}(x)$$

in other words

$$\Delta_{f,1,local}(x) = \max \|f(x) - f(x')\|_1$$

where $\|x - x'\|_H = 1$, *and* $x, x' \in \mathscr{X}$.

To make the difference between local and global sensitivity clear, we want to provide examples for some of the well-known data queries.

Example 1 (Local and Global sensitivity of a median query (Nissim et al., 2007)) *Suppose we have a function* f *which returns the median of input, and we have a set of sorted values defined as*

$$S = \{x_i \in [0, A], \quad x_i \leq x_{i+1} \quad \forall i \in \{1, \ldots, N\}\}.$$

Also, for simplicity, assume that N *is odd as it makes it easier to calculate median. Therefore, for all possible values of* x *in* S *global sensitivity is maximum change between median values of two neighbouring datasets, which is A Nissim et al. (2007). In other words,*

$$\Delta_{Global,f,1} = A.$$

Then, assume $x = (x_1, \ldots, x_m, \ldots, x_N)$ *and* $x_m = \mathcal{T} < A$ *where median is* x_m. *Also,* $x_1, \ldots, x_{m-1} = 0$ *while* $x_{m+1} \ldots x_N = T$. *Then, the local sensitivity on given datasets, as the definition 6 states, is Nissim et al. (2007)*

$$\Delta_{Local,f,1}(x) = \max(x_m - x_{m-1}, x_{m+1} - x_m) = \mathcal{T} < A.$$

Realize that local sensitivity is more efficient for injecting noise compared to the global sensitivity for this case as it introduces less noise.

Example 2 (Local and Global sensitivity of a maximum query (Nissim et al., 2007)) *Suppose we have a function* f *which returns the maximum of input, and we have a set of sorted values defined as*

$$S = \{x_i \in [0, A], \quad x_i \leq x_{i+1} \quad \forall i \in \{1, \ldots, N\}\}.$$

Maximum possible change in the function of maximum query for any $x, x' \in S$ *is* A, *so*

$$\Delta_{Global,f,1} = |f(x) - f(x')| = A.$$

Then, again assume a dataset x where $\mathcal{T} < A$, and the values are

$$x = \{0, \ldots, 0\} \cup \{\mathcal{T}\}.$$

Considering the local sensitivity for the maximum query on given dataset Nissim et al. (2007),

$$\Delta_{Local,f,1}(x) = \max(x_n, x_n - x_{n-1}) = \mathcal{T} < A.$$

Again, local sensitivity is more efficient to use than the global sensitivity for this case.

While the local sensitivity results in less variance than the global sensitivity because of the dependency of the dataset rather than the global population bounds for some cases, directly using local sensitivity instead of a global sensitivity violates the definition 4 (Sun et al., 2020). To prove this, we can assume a case with two neighbors and sorted datasets on set S as

$$x = \{x_1 = \ldots x_{m+1} = 0,\ x_{m+2} = \ldots x_n = \mathcal{T}\},$$
$$x' = \{x_1 = \ldots x_m = 0,\ x_{m+1} \ldots x_n = \mathcal{T}\}.$$

If one uses local sensitivity for adding noise on top of the median, the probability of receiving non-zero answer from D_1 is 0 because $\Delta_{\text{Local,median},1}(x) = \max(x_m - x_{m-1}, x_{m+1} - x_m) = 0$, so noise is 0 and median information is directly reachable. More interestingly, however, $\Delta_{\text{Local,median},1}(x') = \max(x_m - x_{m-1}, x_{m+1} - x_m) = \mathcal{T} \neq 0$, so it is possible to get non-zero answer from private median query on x' as the noise is not 0 anymore Nissim et al. (2007). Hence, the definition 4 is not satisfied. In other words, for small values of δ

$$Pr[f(x) \in S] \nleq e^{\epsilon} \times Pr[f(x') \in S] + \delta$$

Here, there exists a well-designed and safe-to-share upper bound on the local sensitivity called as "smooth bounds". Formally,

Definition 7 (Smooth bounds on local sensitivity (Nissim et al., 2007)) *A function $F : \mathscr{X} \to R^+$ is smooth bound on local sensitivity if*

$$F(x) \leq \epsilon^{\beta} F(y) \quad \textit{and} \quad F(x) \geq \Delta_{Local,f,1}(x) \quad \forall x, y \in \mathscr{X},$$

where $\beta > 0$, and $\|x - y\|_H = 1$.

The smallest F that satisfies definition 7 is $F_{\beta,f}(x) = \max_{y \in D^n}(\Delta_{\text{Local,f,1}}(y).e^{-\beta d(x,y)})$, where $d(x,y)$ stands for hamming distance

between x and y (Nissim et al., 2007). Note that, $F_{\beta,f}(x)$ is called "smooth sensitivity", and it is possible to adjust α and β values to add calibrated noise via privacy mechanisms.

3.2 Privacy mechanisms

Differential privacy mechanisms utilize the sensitivity while perturbating the statistics with random noise to mask the output of a query (Dwork et al., 2014). Some of the well-known noise-adding methods are Laplace, Exponential and Gauss mechanisms, and these are named after the distributions used for perturbation.

The first definition is the Laplace mechanism. The Laplace mechanism employs Laplace distribution and obtains pure differential privacy with $\delta = 0$.

Definition 8 (Laplace mechanism (Dwork et al., 2014)) *Given a function $f : \mathscr{X} \mapsto \mathbb{R}^d$, the randomized algorithm $M(x)$ is $(\epsilon, 0)$-DP if*

$$M(x) = f(x) + V_i \quad \textit{where} \quad V_i \sim Lap(\Delta_{f,1}/\epsilon) \quad \textit{for} \quad i = 1, \ldots, d.$$

Note that $Lap(\Delta_{f,1}/\epsilon) = \frac{1}{2\Delta_{f,1}} exp(\frac{-x}{\Delta_{f,1}/\epsilon})$.

The Laplace mechanism is one of the most popular differential privacy mechanisms as it limits the attacker's ability to obtain sensitive information thanks to the pure differential privacy with $\delta = 0$, and truncating the Laplace distribution allows using interactive queries with bounds (Croft, Sack & Shi, 2022).

Another well-known privacy mechanism is Gaussian mechanism. As the Gaussian distribution has many nice properties, such as adding two Gaussian distributions produces another Gaussian distribution (Dwork et al., 2014), this mechanism is also popular for differential privacy applications. On the other hand, differing from the Laplace mechanism, the Gaussian mechanism satisfies the relaxed version of privacy definition (see definition 4) with $\delta > 0$ using some scaled version of l-2 sensitivity (Dwork et al., 2014).

Definition 9 (Gaussian mechanism (Dwork et al., 2014)) *For $\epsilon \in (0,1)$, randomized algorithm $M(x)$ is (ϵ, δ)-DP if*

$$M(x) = f(x) + V_i \quad \textit{where} \quad V_i \sim \mathcal{N}(0, \sigma^2) \quad \textit{for} \quad i = 1, \ldots, d,$$

where σ is bounded below as

$$\sigma \geq c\Delta_{f,2}/\epsilon, \quad and \quad c^2 > 2ln(1.25/\delta)$$

Note that this definition is only available when $\epsilon \in (0, 1)$. Fortunately, Balle & Wang (2018) extended the definition so that all the ϵ values greater than 0 can be included. They came up with an algorithm (Algorithm 1 in (Balle & Wang, 2018)) that numerically calculates calibrated σ value using the analytical Gaussian cumulative distribution function. The analytical method also alleviates the drawbacks of the classical Gauss mechanism as it performs better when $\epsilon \to 0$, and introduces lower noise compared to the classical method.

For the Gaussian mechanism, it is also possible to come up with another differential privacy definition rather than the classical (ϵ, δ)-DP in definition 4, and this is called Gaussian differential privacy (Dong, Roth & Su, 2022).

The Gaussian differential privacy definition is based on differentially private trade-off functions. One can simply define a trade-off function as

Definition 10 (Trade-off function (Dong et al., 2022)) *Given $M(x)$ and $M(x')$ are the noisy outcomes of the randomized algorithm M, where x and x' are neighboring datasets. Trade-off for $M(x)$ and $M(x')$ is*

$$T(M(x), M(x'))(\alpha) = inf\{\beta_\phi : \alpha_\phi \leq \alpha\}$$

Where β_ϕ and α_ϕ are Type I and Type II errors of hypothesis test of whether the noisy outcome comes from x or x'.

Now, consider another privacy definition based on the trade-off function.

Definition 11 (f-differential privacy (Dong et al., 2022)) *Given f is another trade-off function, the randomized algorithm M is a f-differentially private if*

$$T(M(x), M(x')) \geq f.$$

Hence, f-differential privacy introduces a lower bound for type II error to make D_1 and D_2 indistinguishable. As a result, when the trade-off for any randomized algorithm M using neighboring datasets D_1 and D_2 as input is bounded below to satisfy the definition 11, outcomes of the M become hard to differentiate from each other, so it is differentially private.

Finally, we can define the Gaussian differential privacy combining definition 10 and

definition 11.

Definition 12 (Gaussian differential privacy (Dong et al., 2022)) *Consider randomized algorithm M for neighboring datasets $x, x' \in \mathscr{X}$ as $M(x) = f(x) + V_i$ where $V_i \sim \mathcal{N}(0, \Delta_{f,2}^2/\epsilon^2)$, for $i = 1, \ldots$ Then, M is ϵ-Gussian differentially private (ϵ-GDP) because*

$$T(M(x), M(x')) \geq G_\epsilon \quad \textit{where} \quad G_\epsilon = T(\mathcal{N}(0,1), \mathcal{N}(\epsilon, 1)).$$

Other than ϵ-GDP, there is also a term called "zero-concentrated differential privacy (zCDP)" which aims to minimize the distance between the outputs of a randomized algorithm, and it is also possible to design a Gaussian mechanism satisfying zCDP with a specific noise calibration (Bun & Steinke, 2016). Also, one can also use generalized Gaussian distribution, which provides parametric flexibility to standard normal distribution to model either heavier or lighter tails (Dytso & Shamai, 2018), while designing a Gaussian mechanism (Liu, 2019). However, zero-concentrated differential privacy and generalized Gaussian differential privacy is not closely related to the discussion in this thesis. Therefore, they are not presented in detail.

As Gaussian and Laplace mechanisms, exponential mechanism also plays a crucial role in differential privacy applications since it was designed for highly sensitive outputs that would easily be destroyed with noise (Dwork et al., 2014; McSherry & Talwar, 2007), and it has many applications from optimization to machine learning (Bassily, Smith & Thakurta, 2014; Gopi, Lee & Liu, 2022; Zhu & Yu, 2019). Also, differing from Gaussian mechanism, exponential mechanism satisfies pure differential privacy with $\delta = 0$ like in the Laplace mechanism. Although exponential mechanism is a crucial method for some of the applications, the problem setting in this thesis is not a well-fit for the exponential mechanism. Hence, the definition is not presented in this thesis.

3.3 Post-processing property of differential privacy

Given the privacy mechanisms and their key drivers, sensitivity definitions, one also needs to ensure whether further operations on the private output relaxes the privacy guarantees since most of the inference problems require to work on the outputs of the randomized algorithm repeatedly. Fortunately, differential privacy algorithms

are protected from the further risks of privacy loss, i.e. an attacker can not increase the privacy loss by using the output of a privatized algorithm repeatedly, which is called "post-processing" (Dwork et al., 2014). A formal definition is

Definition 13 (Post-processing (Dwork et al., 2014)) *Consider a randomized algorithm $M_1 : X^n \mapsto O_1$ that satisfies (ϵ, δ)-DP and produces O_1 as an output. Given another randomized algorithm $M_2 : O_1 \mapsto O_2$ that is independent from X^n, $M = (M_1 o M_2)$ is also (ϵ, δ)-DP. To justify, consider x and x' as neighboring datasets, and any event $E \subseteq O_2$. Given another set $S = \{o \in O_1 : M_2(o) \in E\}$,*

$$\begin{aligned} P[M_2(M_1(x)) \in E] &= P[M_1(x) \in S] \\ &\leq e^{\epsilon} P[M_1(x') \in S] + \delta \\ &= e^{\epsilon} P[M_2(M_1(x')) \in E] + \delta \end{aligned}$$

3.4 Composition theorem

This particular feature of differential privacy concerns the combination of private algorithms, and it is especially important as real-life cases may require releasing many sensitive information by running more than one algorithms on same datasets, and one may need to combine various noise mechanisms together to form more capable inference tool (Vadhan & Wang, 2021). However, one may need to be careful while designing composite algorithm since the DP parameters (ϵ, δ) for each individual algorithm are required to be reduced to satisfy (ϵ, δ)-DP on the overall (Dwork et al., 2014). More formally,

Definition 14 (Basic composition (Dwork et al., 2014)) *Given M_t as a (ϵ_t, δ_t)-DP algorithm for $t \in \{1, \ldots, n\}$. Then, $M_{composite} = (M_1, M_2, \ldots, M_n)$ is also a differentially private algorithm with parameters $(\sum_{t=1}^{n} \epsilon_t, \sum_{t=1}^{n} \delta_t)$.*

4. Statistic selection for differential privacy

As mentioned in chapter 3, the data holder adds noise on top of a randomized algorithm that is using sensitive information as input to preserve differential privacy (Dwork, 2008). Then, the analyst is required to work on the noisy output of that specific randomized algorithm. While the differential privacy definition suggests that any randomized algorithm satisfies the desired protection if the noise is well-designed, the informativeness and utility of the study conducted by the analyst are also crucial in privacy-preserving data analytics (Oberski & Kreuter, 2020). Hence, it is important to discuss which function of the data should be used while releasing the noisy output so that the utility of the study is maximized while not sacrificing privacy. As we discuss in the below, one way of choosing best statistic is using Fisher information, a well-known mathematical measure of information. Discussion in this section is published at Alparslan & Yıldırım (2022), a journal article at Statistics and Computing in 2022.

4.1 Notation and privacy setting

Consider a data setting as $X_1, \ldots, X_n \sim P_\theta$, where P_θ is any distribution with $\theta \in \Theta$, and each X_i contains sensitive information of an individual. Also, for the case of batch sharing, we introduce a statistic of the data as $S_n(X_{1:n})$, where S_n maps the input data of n dimension to real output as $S_n : \mathcal{X}^n \mapsto \mathbb{R}^{d_s}$ with $d_s \geq 1$. To protect sensitive information, we design to release the statistic S_n with some calibrated noise as

$$Y = S_n(X_{1:n}) + V \quad \text{where} \quad V \sim \mathcal{P}_{\epsilon, S_n}. \tag{4.1}$$

$\mathcal{P}_{\epsilon,S_n}$ is a distribution for privacy mechanism whose parameters are determined by ϵ and sensitivity of S_n. Additionally, it is also possible to design a sequential releasing case where each X_i is shared with some noise as

$$Y_i = s(X_i) + V \quad \text{where} \quad V \sim \mathcal{P}_{\epsilon,s}. \tag{4.2}$$

4.2 Selection based on Fisher information

One way of measuring the informativeness of a certain statistic is the Fisher information, which captures the amount of information that the obtained data includes about the desired parameter Chao, Sally Ward & Ober (2016). Thanks to the Bernstein-von Misses theorem, posterior distribution of a parameter is approximately normally distributed when the number of observations converges to the infinity, and covariance matrix of this distribution is inversely related to the Fisher information (definition 15) of that certain parameter (Kleijn & Van der Vaart, 2012; Vaart, 1998). Then, owing to the Bernstein-von Misses theorem, one can effectively utilize Fisher information in the Bayesian estimation setting as higher Fisher information means less variated estimator in the long run. In the following parts, we define/derive Fisher information for various cases combining various noise mechanisms and candidate statistics.

Definition 15 (Fisher information (Schervish, 1995))

$$\begin{aligned} F(\theta) &= \mathbb{E}\left[-\frac{\partial^2 log p_{\epsilon,S_n}(Y|\theta)}{\partial\theta\partial\theta^T}\right], \\ &= \mathbb{E}\left[\gamma_{\epsilon,S_n}(\theta;Y)\gamma_{\epsilon,S_n}(\theta;Y)^T\right], \end{aligned}$$

where

$$\begin{aligned} &\gamma_{\epsilon,S_n} = \frac{log p_{\epsilon,S_n}(Y|\theta)}{\partial\theta}, \\ &p_{\epsilon,S_n}(Y = y|\theta) = \int p_{\epsilon,S_n}(y|x_{1:n}) \prod_{i=1}^{n} p(x_i|\theta) dx_{1:n}. \end{aligned}$$

Under some regularity conditions;

2.1 The partial derivative of $P(Y|\theta)$ exists almost everywhere.

2.2 The integral of $P(Y|\theta)$ can be differentiated w.r.t. θ.

2.3 The support of $P(Y|\theta)$ doesn't depend on θ.

Calculation of Fisher information given in definition 15 is closely contingent upon

the noise distribution, and the type of the statistic in equations (4.1), (4.2).

While using additive statistic for the differential privacy mechanism allows tractable calculation thanks to the normal approximation, it is also possible to use non-Gaussian noise or non-additive statistics where the Fisher information is not analytically available. In this case, one can efficiently utilize Monte Carlo integration in equation (2.3) to approximate the value numerically.

4.2.1 Fisher information with additive statistic and Gaussian noise

The first case is built upon additive statistic and Gaussian noise where the Fisher information is analytically available. In detail, consider

$$\begin{aligned} S_n(X_{1:n}) &= \frac{1}{n}\sum_{i=1}^{n} s(X_i), \\ Y &= S_n(X_{1:n}) + V \quad \text{where} \quad V \sim \mathcal{N}(0, \frac{\Delta_{s,2}^2}{n^2\epsilon^2} I), \end{aligned} \tag{4.3}$$

where $\Delta_{s,2}^2$ is l-2 sensitivity of function s. Note that this mechanism doesn't satisfy (ϵ, δ)-DP in definition 4, instead it is designed for ϵ-GDP in definition 12.

As $S_n(X_{1:n})$ is simply an average operation, it asymptotically follows the normal distribution by the central limit theorem (CLT). Following the notation from Bernstein & Sheldon (2018), we define mean and covariance of $s(X)$ as

$$\mu_s(\theta) = \mathbb{E}_\theta\left[s(X)\right], \qquad \Sigma_\theta = \mathrm{Var}_\theta\left[s(X)\right].$$

Then, the normal approximation for $S_n(X_{1:n})$ is

$$S_n(X_{1:n}) \sim \mathcal{N}(\mu_s(\theta), \Sigma_\theta/n).$$

Using equation (4.3), we can derive the distribution of Y as

$$Y \sim \mathcal{N}(\mu_s(\theta), \Sigma_\theta/n + \frac{\Delta_{s,2}^2}{n^2\epsilon^2} I).$$

Given the fact that Y has a well-defined Gaussian distribution, the task of deriving Fisher information of θ on the distribution of Y is relatively straightforward as Fisher information for multivariate normal distribution is available in a closed form (Malagò

& Pistone, 2015). Considering the transformation $\theta \mapsto \left[\mu_s(\theta), \Sigma_\theta/n + \frac{\Delta_{s,2}^2}{n^2\epsilon^2} I\right]$,

$$[F(\theta)]_{i,j} = \frac{\partial \mu_s(\theta)^T}{\partial \theta_i} H_{s,\epsilon,n}(\theta)^{-1} \frac{\partial \mu_s(\theta)}{\partial \theta_j} + \frac{\mathrm{tr}(G)}{2}$$

where $H_{s,\epsilon,n}(\theta) := \frac{\Sigma_s(\theta)}{n} + \frac{\Delta_{s,2}^2}{n^2\epsilon^2} I$ is the covariance of Y and

$$G = \frac{1}{n^2}\left(H_{s,\epsilon,n}(\theta)^{-1} \frac{\partial \Sigma_s(\theta)}{\partial \theta_i} H_{s,\epsilon,n}(\theta)^{-1} \frac{\partial \Sigma_s(\theta)}{\partial \theta_j}\right).$$

At this point, we want to provide some examples to demonstrate the Fisher information for some common inference cases.

Example 3 (Mean of the normal distribution) *Consider the distribution of $X \sim \mathcal{N}(\theta, 1)$, and X takes values in between $(0, A)$. Technically, this means that X has truncated normal distribution, but for the sake of tractability of the calculation, we choose θ values far away from A so that the data bounds has negligible effects on the calculations. The reason we introduce bounds for the data is that we need to derive l-2 sensitivity for $s(X)$. Also, consider $s(X) = x^a$ where a is an odd integer, we focus on $a = \{1, 3\}$ for this example, and $\Delta_{s,2} = A^a$. Using higher moments of the normal distribution, mean and variance values are*

$$\mu_s(\theta), \Sigma_s(\theta) = \begin{cases} \theta, 1 & a = 1 \\ \theta^3 + 3\theta, 9\theta^4 + 36\theta^2 + 15 & a = 3. \end{cases}$$

Fortunately, taking the derivation w.r.t θ is straightforward as

$$\frac{\partial \mu_s(\theta)}{\partial \theta}, \frac{\partial \Sigma_s(\theta)}{\partial \theta} = \begin{cases} 1, 0 & a = 1 \\ 3\theta^2 + 3, 36\theta^3 + 72\theta & a = 3. \end{cases}$$

In the following figure, we compare $F(\theta)$ when $a = 1$ and $a = 3$ for $\epsilon = 1$ and $\epsilon = \infty$. We use $n = 100$ and $A = 10$, and make a comparison for various θ values.

According to the figure 4.1, while $s(x) = x$ is more informative for non-private case ($\epsilon = \infty$), $s(x) = x^3$ becomes more informative for higher values of θ under tight privacy constraints ($\epsilon = 1$).

Example 4 (Variance of the normal distribution) *This time, consider $X \sim P_\theta = \mathcal{N}(0, \theta)$, and take $X \in [-A, A]$. Using the same assumption with example 3, effect of A on the calculations are negligible as it is a large number. For $s(X) = |x|^a$,*

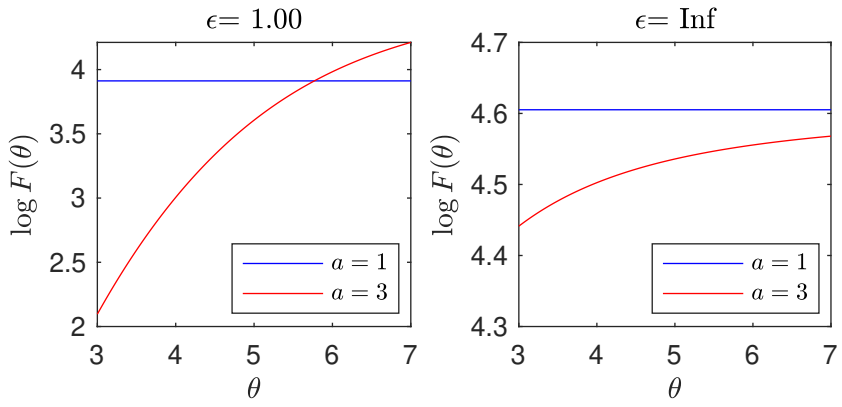

Figure 4.1 $F(\theta)$ for the mean parameter of $\mathcal{N}(\theta,1)$ when $s(x)=x^a$. Left: $\epsilon=1$, Right: $\epsilon=\infty$ (non-private case).

mean and covariance values are approximately

$$\mu_s(\theta)=\mathbb{E}_{P_\theta}(|x|^a)=(2\theta)^{a/2}\frac{1}{\sqrt{\pi}}\Gamma\left(\frac{a+1}{2}\right),$$
$$\Sigma_s(\theta)=Var_{P_\theta}(|x|^a)=(2\theta)^a\left[\frac{1}{\sqrt{\pi}}\Gamma\left(\frac{2a+1}{2}\right)-\frac{1}{\pi}\Gamma^2\left(\frac{a+1}{2}\right)\right].$$

Taking the derivative w.r.t θ is also easy-to-handle:

$$\frac{\partial\mu_s(\theta)}{\partial\theta}=\frac{2^{\frac{a}{2}}a}{2}\theta^{(a-2)/2}\frac{1}{\sqrt{\pi}}\Gamma\left(\frac{a+1}{2}\right),$$
$$\frac{\partial\Sigma_s(\theta)}{\partial\theta}=2^a a\theta^{a-1}\left[\frac{1}{\sqrt{\pi}}\Gamma\left(\frac{2a+1}{2}\right)-\frac{1}{\pi}\Gamma^2\left(\frac{a+1}{2}\right)\right].$$

Then, we take $A=100$, and $n=100$. Like in the example 3 for various θ values with epsilon $=1$ and $\epsilon=\infty$ can be seen in the figure 4.2.

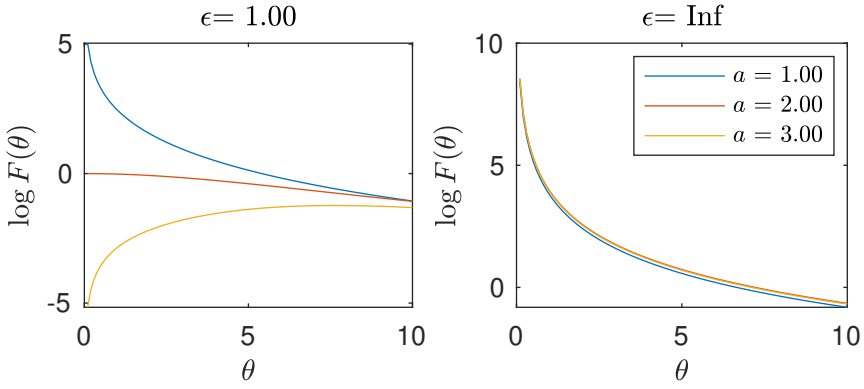

Figure 4.2 $F(\theta)$ for the variance parameter of $\mathcal{N}(0,\theta)$ when $s(x)=|x|^a$. Left: $\epsilon=1$, Right: $\epsilon=\infty$ (non-private case).

According to figure 4.2, $s(X)=x^2$ is more informative for all θ values when privacy is not the concern. On the other hand, $s(X)=|x|$ takes the lead when $\epsilon=1$.

Example 5 (Width of the uniform distribution) *Differing from the normal distribution, now consider uniform distribution. In other words,* $X \sim P_\theta = Unif(-\theta,\theta)$ *with* $X \in [-A,A]$*. Note that, Fisher information for the width parameter of uniform distribution doesn't exist as it violates regularity conditions 1 and 3 in definition 15, but it does exist for marginal distribution of* Y *given differentiable* $\mu_s(\theta),\Sigma_s(\theta)$ *since support of* $P(Y|\theta)$ *doesn't depend on* θ *because of the normal approximation.*

Once we use $s(X) = |x|^a$ *as in the previous example, the mean and covariance values become*

$$\mu_s(\theta) = \frac{\theta^a}{a+1}, \quad \Sigma_s(\theta) = \frac{\theta^{2a}a^2}{(a+1)^2(2a+1)}.$$

The derivatives w.r.t. θ *are:*

$$\frac{\partial \mu_s(\theta)}{\partial \theta} = \frac{a}{a+1}\theta^{a-1},$$
$$\frac{\partial \Sigma_s(\theta)}{\partial \theta} = \frac{2a^3}{(a+1)^2(2a+1)}\theta^{2a-1}.$$

Given $\mu_s(\theta),\Sigma_s(\theta)$ *and their derivatives, figure 4.3 compares* $F(\theta)$ *with different* a *and* ϵ *values using* $n = 100$ *and* $A = 10$*.*

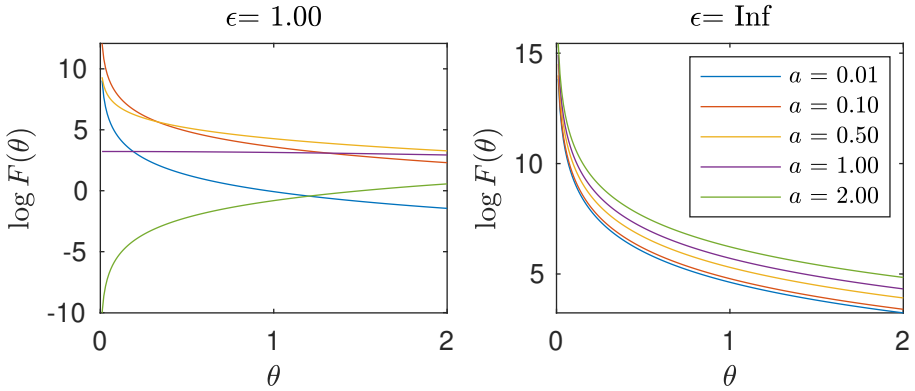

Figure 4.3 $F(\theta)$ for the width parameter of Unif$(-\theta,\theta)$ when $s(x) = |x|^a$. Left: $\epsilon = 1$, Right: $\epsilon = \infty$ (non-private case).

Using figure 4.3, one can obtain that while $s(X) = x^2$ *is the best for non-private case,* $s(X) = \sqrt{|x|}$ *and* $s(X) = |x|^{0.1}$ *dominates the other statistics when the privacy level is increased to* $\epsilon = 1$*.*

This example demonstrates a promising fact for the mentioned statistic selection model as it applies even for the well-known cases where the Fisher information doesn't exist properly.

4.2.2 Fisher information with additive statistic and non-Gaussian noise

Until now, the Gaussian mechanism enabled us to perform analytical comparison without estimating the value of $F(\theta)$. However, when the mechanism changes and the noise becomes non-Gaussian things get complicated. More specifically, the probability distribution of $p_{\epsilon,S_n}(Y=y|\theta)$ in definition 15 is not tractable anymore, hence the derivative of logarithm w.r.t. θ would also be analytically incalculable although $S_n(X_{1:n})$ is still approximately Gaussian. Fortunately, Monte Carlo estimators can be efficiently employed to approximate the value of $F(\theta)$ numerically.

Let us first define the problem setting which is similar to the previous case in section 4.2.1 with only a difference as

$$\begin{aligned} &S_n(X_{1:n}) = \frac{1}{n}\sum_{i=1}^{n} s(X_i), \\ &Y = S_n(X_{1:n}) + V \quad \text{where} \quad V \sim \text{non-Gaussian distribution.} \end{aligned} \tag{4.4}$$

Since the additive statistic still allows us to use normal approximation for the distribution of $S_n(X_{1:n})$, we define a new random variable as $U = S_n(X_{1:n})$. Then, we have

$$f_{S_n}(U=u|\theta) \sim \mathcal{N}(u;\, \mu_s(\theta), \Sigma_s(\theta)/n). \tag{4.5}$$

Additionally, we can define $g_{\epsilon,S_n}(Y|U)$ as the conditional distribution of Y given U, so the marginal density of Y given θ calculated at $Y=y$ and $U=u$ is

$$\begin{aligned} p_{\epsilon,S_n}(y|\theta) &= \int p_{\epsilon,S_n}(y|u,\theta) f_{S_n}(u|\theta) du, \\ &= \int g_{\epsilon,S_n}(y|u) f_{S_n}(u|\theta) du. \end{aligned} \tag{4.6}$$

This is because $p(y|u,\theta) = \frac{p(y,u,\theta)}{p(u,\theta)} = \frac{p(y|u)p(u|\theta)p(\theta)}{p(u|\theta)p(\theta)} = p(y|u)$.

By combining equation (4.6) and the Fisher's identity Douc, Moulines & Stoffer (2013),

$$\begin{aligned} \gamma_{\epsilon,S_n}(\theta;y) = \frac{\partial \log p_{\epsilon,S_n}(Y|\theta)}{\partial\theta} &= \int \frac{\partial \log p(y,u|\theta)}{\partial\theta} p(u|y,\theta) du, \\ &= \int \frac{\partial \left[\log f_{S_n}(u|\theta) + \log g_{\epsilon,S_n}(y|u)\right]}{\partial\theta} p(u|y,\theta) du \\ &= \int \frac{\partial \log f_{S_n}(u|\theta)}{\partial\theta} p(u|y,\theta) du, \end{aligned} \tag{4.7}$$

where the integral is taken with respect to the posterior distribution $p(u|y,\theta) \propto$

$f_{S_n}(u|\theta)g_{\epsilon,S_n}(y|u)$. Note that $p(y,u|\theta) = f_{S_n}(u|\theta)g_{\epsilon,S_n}(y|u)$ since

$$\begin{aligned} p(y,u|\theta) &= \frac{p(y|u,\theta)p(u|\theta)p(\theta)}{p(\theta)} \\ &= p(y|u)p(u|\theta) \\ &= f_{S_n}(u|\theta)g_{\epsilon,S_n}(y|u). \end{aligned}$$

Accurate estimation of the score vector in equation (4.7) is vital as it is a key element of $F(\theta)$ as in definition 15. For this purpose, it is possible to employ several methods to estimate integral, namely one can use importance sampling, rejection-based sampling or approximate sampling (via MCMC approaches).

A Monte Carlo estimator of $F(\theta)$ based on the self-normalized importance sampling (See chapter 2, section 2.2) is given in Algorithm 6.

Algorithm 6 Monte Carlo estimation of $F(\theta)$ for (4.4) - normal approximation for $f_{S_n}(u|\theta)$.

Input: θ: parameter; n: data size; N, M: Monte Carlo parameters
Output: $\widehat{F(\theta)}$: Estimate of $F(\theta)$
for $i = 1,\dots,N$ **do**
  Sample $y^{(i)} \sim p_{\epsilon,S_n}(y|\theta)$
  for $j = 1,\dots,M$ **do**
    Sample $u^{(j)} \sim q_\theta(\cdot)$, calculate $w_j = f_{S_n}(u^{(j)}|\theta)g_{\epsilon,S_n}(y^{(i)}|u^{(j)})/q_\theta(u^{(j)})$ using (4.5).
  Calculate $\widehat{\gamma}_{\epsilon,S_n}(\theta;y^{(i)}) = \sum_{j=1}^{M} \frac{\partial \log f_{S_n}(u^{(j)}|\theta)}{\partial\theta} \frac{w_j}{\sum_{j'=1}^{N} w_{j'}}$ using (4.5).
return $\widehat{F(\theta)} = \frac{1}{N}\sum_{i=1}^{N} \widehat{\gamma}_{\epsilon,S_n}(\theta;y^{(i)})\widehat{\gamma}_{\epsilon,S_n}(\theta;y^{(i)})^T$.

4.2.3 Fisher information based on the true marginal distribution

As we mentioned above, Algorithm 6 employs normal approximation for the statistic $S_n(X_{1:n})$ using CLT (See equation (4.4)). On the other hand, normal approximation may not be viable for $S_n(X_{1:n})$ as moments $\mu_s(\theta), \Sigma_s(\theta)$ may not be tractable, or one can prefer using non-additive statistic, like median or maximum, for $S_n(X_{1:n})$ as in

$$S_n(X_{1:n}) = \max_i s(X_i) \quad \text{or} \quad S_n(X_{1:n}) = \text{median}(s(X_i)).$$

Non-additive statistic for $S_n(X_{1:n})$ not only affects conditional distribution of $p(S_n(X_{1:n})|\theta)$, but sensitivity calculation for privacy mechanism also differs as we

discussed in chapter 3, section 3.1. More specifically, for some non-additive statistics smooth sensitivity provides safe and more efficient way of perturbation as in definition 7. As an example, smooth sensitivity with the Laplace mechanism is obtained by a noise distribution

$$V \sim \text{Laplace}(\Delta^{\text{smooth}}_{S_n,\beta}/\alpha)$$
$$Y = S_n(X_{1:n}) + V$$

While it is still possible to estimate score vector as before, the distribution of $S_n(X_{1:n})$ is not tractable anymore, so notation differs from the previous algorithm that exploits $U = S_n(X_{1:n}) \sim \mathcal{N}$. This time, there is no particular benefit of using hidden variable U, instead we resort $X_{1:n}$. Then, the method exploits observations of $x^{(j)}_{1:n}$ from the population distribution rather than $u^{(j)}$ for the calculation of smooth sensitivity. As a result of elimination of hidden variable U from the model, the Fisher score vector can be derived using the exact marginal distribution in definition 15 and the Fisher's identity (Douc et al., 2013), which is

$$\begin{aligned}
\gamma_{\epsilon,S_n}(\theta; y) = \frac{\partial \log p_{\epsilon,S_n}(Y|\theta)}{\partial\theta} &= \int \frac{\partial \log p(y, x_{1:n}|\theta)}{\partial\theta} p(x_{1:n}|y,\theta) dx_{1:n}, \\
&= \int \frac{\partial \left[\log p(x_{1:n}|\theta) + \log p(y|x_{1:n})\right]}{\partial\theta} p(x_{1:n}|y,\theta) dx_{1:n}, \\
&= \int \frac{\partial \log p(x_{1:n}|\theta)}{\partial\theta} p(x_{1:n}|y,\theta) dx_{1:n}, \\
&= \int (\sum_{i=1}^{n} \frac{\partial \log p(x_i|\theta)}{\partial\theta}) p(x_{1:n}|y,\theta) dx_{1:n}, \qquad (4.8)
\end{aligned}$$

where $p(x_{1:n}|y,\theta) \propto p_{\epsilon,S_n}(y|x_{1:n})$.

Note that, $p_{\epsilon,S_n}(y|x_{1:n})$ is density function V that is evaluated at $y - S_n(x_{1:n})$ from equation (4.4).

Such an integral can be numerically calculated using self-normalized importance sampling (See chapter 2, section 2.2) with samples of $X_{1:n}$ as in Algorithm 7.

4.2.4 Fisher information with sequential release

Until now, we discussed about how Fisher information is defined for summary statistics (additive and non-additive) considering a single release $S_n(X_{1:n})$ with a cali-

Algorithm 7 Monte Carlo estimation of $F(\theta)$ - exact marginal distribution

Input: θ: parameter; n: data size; N, M: Monte Carlo parameters

Output: $\widehat{F(\theta)}$: Estimate of $F(\theta)$

for $i = 1, \ldots, N$ **do**

  Sample $y^{(i)} \sim p_{\epsilon,S_n}(y|\theta)$

  for $j = 1, \ldots, M$ **do**

    for $t = 1, \ldots, n$ **do**

      Sample $x_t^{(j)} \sim p(x|\theta)$.

    Set $w_j = p_{\epsilon,S_n}(y^{(i)}|x_{1:n}^{(j)})$.

  Calculate $\widehat{\gamma}_{\epsilon,S_n}(\theta; y^{(i)}) = \sum_{j=1}^{M} \left(\sum_{t=1}^{n} \frac{\partial \log p(x_t^{(j)}|\theta)}{\partial \theta} \right) \frac{w_j}{\sum_{j'=1}^{M} w_{j'}}$.

return $\widehat{F(\theta)} = \frac{1}{N} \sum_{i=1}^{N} \widehat{\gamma}_{\epsilon,S_n}(\theta; y^{(i)}) \widehat{\gamma}_{\epsilon,S_n}(\theta; y^{(i)})^T$.

brated noise. However, it is also possible to design a sequential release mechanism where each $s(X_i)$ is altered with noise as in equation (4.2), which is sometimes referred as local model Kasiviswanathan, Lee, Nissim, Raskhodnikova & Smith (2008). In a sequential model, an ϵ-DP privacy with the Laplace mechanism looks like

$$Y = s(X_i) + V \quad \text{where} \quad V \sim \text{Laplace}(\Delta_{s,1}/\epsilon). \tag{4.9}$$

Note that, in equation (4.9), it is obvious that local model adds more noise because of the lack of n in the denominator.

Revisiting the marginal distribution of Y, this time we can write it for each Y_i whose probability density is

$$p_{\epsilon,s}(y|\theta) = \int p(x|\theta) g_{\epsilon,s}(y|s(x)) dx, \tag{4.10}$$

where $g_{\epsilon,s}(y|s(x))$ is the probability density of Y given $s(X)$ calculated at y, which is equivalent of calculating the probability density of $P_{\epsilon,s}$ in equation (4.2) at $y - s(x)$. We prefer integration over x instead of on $s(x)$ as $p(x|\theta)$ is assumed to be available in the problem setting.

Similar to the equation (4.8), one can calculate Fisher information as

$$\begin{aligned} \gamma_{\epsilon,S_n}(\theta; y) = \frac{\partial \log p_{\epsilon,S_n}(Y|\theta)}{\partial \theta} &= \int \frac{\partial \log p(y, x|\theta)}{\partial \theta} p(x|y, \theta) dx, \\ &= \int \frac{\partial \left[\log p(x|\theta) + \log p(y|x) \right]}{\partial \theta} p(x|y, \theta) dx, \\ &= \int \frac{\partial \log p(x|\theta)}{\partial \theta} p(x|y, \theta) dx, \end{aligned} \tag{4.11}$$

where $p(x|y, \theta) \propto p(x|\theta) g_{\epsilon,s}(y|s(x))$.

Algorithm 8 demonstrates the application of self-normalized importance sampling for the sequential release case.

Algorithm 8 Monte Carlo estimation of $F(\theta)$ for (4.2)

Input: θ: parameter; n: data size; N, M: Monte Carlo parameters

Output: $\widehat{F(\theta)}$: Estimate of $F(\theta)$

for $i = 1, \ldots, N$ **do**

  Sample $y^{(i)} \sim p_{\epsilon,s}(y|\theta)$

  for $j = 1, \ldots, M$ **do**

    Sample $x^{(j)} \sim q_\theta(x)$ and calculate $w_j = p(x^{(j)}|\theta) g_{\epsilon,s}(y^{(i)}|s(x^{(j)}))/q_\theta(x^{(j)})$.

  Calculate $\widehat{\gamma}_{\epsilon,s}(\theta; y^{(i)}) = \sum_{j=1}^{M} \frac{\partial \log p(x^{(j)}|\theta)}{\partial \theta} \frac{w_j}{\sum_{j'=1}^{M} w_{j'}}$.

return $\widehat{F(\theta)} = \frac{n}{N} \sum_{i=1}^{N} \widehat{\gamma}_{\epsilon,S_n}(\theta; y^{(i)}) \widehat{\gamma}_{\epsilon,S_n}(\theta; y^{(i)})^T$.

To further clarify local model with sequential release and compare with the batch release we want to give an example setting including three separate scenarios based on binary randomized responses.

Example 6 (Binary responses) *Consider a Bernoulli distribution for P_θ, that is population distribution of i.i.d X_i's from $i = 1, \ldots, n$ as*

$$P_\theta(X_i) = \begin{cases} 1, & \theta \\ 0, & 1-\theta. \end{cases}$$

In a non-private setting, maximum likelihood estimator for θ is $\bar{X}$. However, for the private estimation, it may be different. Therefore, we want to calculate Fisher information for the following three mechanisms.

3.1 We consider releasing $Y_1, \ldots, Y_n \in \{0,1\}$, where

$$Y_i = \begin{cases} X_i, & \frac{e^\epsilon}{1+e^\epsilon} \\ 1 - X_i & \frac{1}{1+e^\epsilon} \end{cases}$$

This mechanism safely provides ϵ-DP. To justify, consider two neighboring datasets $D_1 = \{x_1, \ldots, x_{n-1}, 1\}$ and $D_2 = \{x_1, \ldots, x_{n-1}, 0\}$, and a randomized algorithm $M : \{0,1\} \mapsto S \in \{0,1\}$. After fixing a case for Y, due to the independence, we can write that

$$P(Y = 1|D_1) = p(y = 1|x_n = 1) \prod_{i=1}^{n-1} p(y_i|x_i),$$

$$P(Y = 1|D_2) = p(y = 1|x_n = 0) \prod_{i=1}^{n-1} p(y_i|x_i),$$

where $p(y=1|x_n=1)=\frac{e^\epsilon}{1+e^\epsilon}$ *and* $p(y=1|x_n=0)=\frac{1}{1+e^\epsilon}$ *. Hence,*

$$\frac{P[M(D_1)=1\in S]}{P[M(D_2)=1\in S]}=\frac{P(Y=1|D_1)}{P(Y=1|D_2)}=e^\epsilon\leq e^\epsilon.$$

This is the end of justification as the proportion above demonstrates that this mechanism satisfies definition 4 with $\delta=0$*.*

Additionally, we know that

$$\tau:=P(Y=1)=\frac{\theta e^\epsilon}{1+e^\epsilon}+\frac{1-\theta}{1+e^\epsilon}=\frac{\theta e^\epsilon+(1-\theta)}{1+e^\epsilon},$$
$$\log p(y|\theta)=y\ln\tau+(1-y)\ln(1-\tau).$$

Therefore, Fisher information of $Y_1,\ldots,Y_n$ *in this case is*

$$F_1(\theta)=n\mathbb{E}\left[-\frac{\partial^2\log p(Y|\theta)}{\partial\theta^2}\right]=\frac{n\alpha^2}{\tau(1-\tau)},\quad \textit{where}\quad \alpha=\frac{e^\epsilon-1}{e^\epsilon+1}.$$

3.2 Another alternative to releasing Y_i*'s as above, one can also consider disclosing* $\hat{\theta}_2=\bar{Z}=\frac{1}{n}\sum_{i=1}^N Z_i$*, where*

$$Z_i=X_i+V_i\quad \textit{given that}\quad V_i\sim\mathcal{N}(0,1/\epsilon^2).$$

Note that, each X_i *is perturbated with a noise drawn from normal distribution with a variance of* $1/\epsilon^2$*. According to the Gaussian mechanism in definition 12, this type of noise satisfies* ϵ*-GDP as* $\Delta_{X,2}=1$*. Hence, owing to the post-processing property of differential privacy in definition 13, the mechanism releasing* $\hat{\theta}_2$ *is also differentially private.*

By the central limit theorem, we can say that $\hat{\theta}_2\sim\mathcal{N}(\theta,\theta(1-\theta)/n+1/(\epsilon^2 n))$*, and after some algebraic operations Fisher information is approximately*

$$F_2(\theta)=\mathbb{E}\left[-\frac{\partial^2\log p(\hat{\theta}_2|\theta)}{\partial\theta^2}\right]\approx\frac{n(\theta(1-\theta)+1/\epsilon^2)+(1-2\theta)^2}{[\theta(1-\theta)+1/\epsilon^2]^2}.$$

3.3 Finally, as we discussed in previous sections with particular focus on summary statistics, this time consider releasing noisy average $\hat{\theta}_3$ *where*

$$\hat{\theta}_3=\bar{X}+V,\quad \textit{where}\quad V\sim\mathcal{N}(0,1/(n^2\epsilon^2)).$$

Note that, this mechanism adds smaller noise comparing with the previous case, which results in improvement in the Fisher information as in the follow-

ing equation.

Similarly, $\hat{\theta}_3 \sim \mathcal{N}(\theta, \theta(1-\theta)/n + 1/(\epsilon^2 n^2))$*, and Fisher's information is approximately*

$$F_3(\theta) = \mathbb{E}\left[-\frac{\partial^2 \log p(\hat{\theta}_3|\theta)}{\partial \theta^2}\right] \approx \frac{n(\theta(1-\theta)+1/(\epsilon^2 n)) + (1-2\theta)^2}{[\theta(1-\theta)+1/(\epsilon^2 n)]^2}.$$

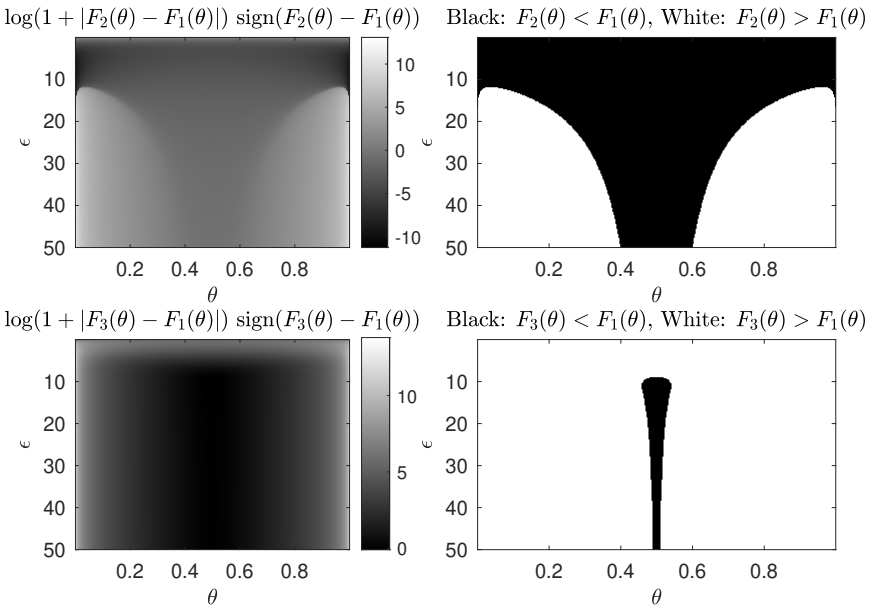

Figure 4.4 Comparison among $F_1(\theta)$, $F_2(\theta)$, $F_3(\theta)$.

Figure 4.4 compares Fisher information for these three cases using $n = 100$*. It demonstrates that for the small* ϵ *values releasing randomized responses(*$F_1(\theta)$*) is better than releasing the average of noisy responses (*$F_2(\theta)$*), while in the same* ϵ *regime releasing the noisy average (*$F_3(\theta)$*) is better than randomized responses (*$F_1(\theta)$*).*

5. Bayesian inference with differential privacy

So far, differential privacy and Bayesian computation has been extensively studied by many researchers as Bayesian estimation can be efficiently utilized without violating privacy constraints and exceeding the privacy budget (Dimitrakakis, Nelson, Zhang, Mitrokotsa & Rubinstein, 2017). Posterior sampling under the private setting is easy-to-handle as it is similar to non-private setting and satisfies differential privacy under some necessary and sufficient conditions (Dimitrakakis et al., 2017). More on that, Wang et al. (2015) shows that Bayesian model with bounded likelihood is consistent and differentially private. Hence, there exists variety of research products spanning wide range from optimization (Kharkovskii, Dai & Low, 2020; Kusner, Gardner, Garnett & Weinberger, 2015; Li, Chen, Liu & Carin, 2019; Ryffel, Bach & Pointcheval, 2022) to the machine learning problems (Bernstein & Sheldon, 2019; Ju, Awan, Gong & Rao, 2022b; Wang, 2018; Zhang, Bu, Chen & Long, 2021) combining Bayesian inference with differential privacy.

While the field is extremely active with many works, some of them, especially the ones using MCMC methods for Bayesian inference, particularly concerns this thesis as we are more focused on Monte Carlo methods for the differential privacy applications. In that sense, Heikkilä et al. (2019) proposes a generalized scheme for MCMC application in differential privacy setting using Renyi differential privacy, which is a form of privacy based on distributional distance, and Wang et al. (2015) focuses on stochastic gradient Langevin dynamics, a method for solving optimization problems. However, both of them employed non-exact MCMC where the samples converges to the posterior distribution asymptotically. On the contrary, Yıldırım & Ermiş (2019) proposes MCMC scheme using penalty algorithm to target posterior distribution under differential privacy. As stationary distribution of penalty algorithm remains exact target posterior distribution, their method demonstrates an application of exact MCMC. Then, Räisä et al. (2021) develops Hamiltonian Monte Carlo based on the exact MCMC in Yıldırım & Ermiş (2019).

One common property of the works mentioned above is that they rely on continuous noise perturbation as they release noisy outputs in every iteration. However, this

may not be possible all the time due to the practical concerns. Fortunately, differing from those works, one can also share data once and for all. For instance, Foulds, Geumlek, Welling & Chaudhuri (2016) uses Gibbs sampling to sample new value of the parameter using data that is privatized only once. Ju et al. (2022b) also uses privatized data in a similar way for Gibbs sampling but with a purpose of linear regression. Gong (2019) employs EM algorithm for maximum likelihood estimator within the Bayesian framework.

In this chapter, firstly, we present Bayesian inference methods developed to work harmonically with the statistic selection methods mentioned in chapter 4. Then, independent from the statistic selection discussion, we change our focus to the Bayesian linear regression problems by proposing various algorithms based on MCMC methods.

5.1 Bayesian estimation after statistic selection

The idea of sampling comes from approximate Bayesian computation in Gong (2019), while the EM algorithm mentioned previously may not be feasible when exact posterior expectations don't exist. On the other hand, MCMC methods in chapter 2 can be easily combined with statistic selection with Fisher information that was presented in chapter 4. Hence, this section is resorted for the proposed MCMC algorithms. Note that discussion in this section is a part of the same publication at Alparslan & Yıldırım (2022).

In detail, for the batch and sequential settings described in equations (4.1),(4.11) respectively, it is possible to write posterior distribution for the target parameter θ as,

$$\begin{aligned} p(\theta|y) &\propto \eta(\theta) p_{\epsilon,S_n}(y|\theta), && \text{(batch setting)} \\ p(\theta|y_{1:n}) &\propto \eta(\theta) \prod_{i=1}^{n} p_{\epsilon,s}(y_i|\theta), && \text{(sequential setting)} \end{aligned}$$

where $\eta(\theta)$ represents the prior density of θ, and $p_{\epsilon,S_n}(y|\theta)$, $p_{\epsilon,s}(y_i|\theta)$ are likelihoods of the observations. Also note that, the likelihood for the batch setting was defined

in the previous chapter as

$$p_{\epsilon,S_n}(y|\theta) = \int p_{\epsilon,S_n}(y|x_{1:n}) \prod_{i=1}^{n} p(x_i|\theta) dx_{1:n}, \qquad \text{(batch setting)}$$

$$p_{\epsilon,s}(y|\theta) = \int p(x|\theta) g_{\epsilon,s}(y|s(x)) dx, \qquad \text{(sequential setting)}$$

Following algorithms uses above posterior distributions to cover all the cases of statistic selection i.e. additive/non-additive statistics and Gaussian/non-Gaussian noises.

5.1.1 MH for additive statistic and Gaussian noise

To design an inference method for this part, one can also exploits the coherence between normal approximation and the Gaussian noise as in section 4.2.1 in chapter 4. Remember that

$$S_n(X_{1:n}) = \frac{1}{n} \sum_{i=1}^{n} s(X_i),$$

$$Y = S_n(X_{1:n}) + V, \quad \text{where} \quad V \sim \mathcal{N}(0, \frac{\Delta_{s,2}^2}{n^2 \epsilon^2} I).$$

and using the normal approximation for $S_n(X_{1:n})$, it is easy to obtain that

$$Y \sim \mathcal{N}(\mu_s(\theta), \Sigma_\theta / n + \frac{\Delta_{s,2}^2}{n^2 \epsilon^2} I).$$

Therefore, posterior density of θ given y can be identified as

$$\hat{p}_{\epsilon,S_n}(\theta|y) \propto \eta(\theta) \mathcal{N}(y; \mu_s(\theta), H_{s,\epsilon,n}(\theta)). \tag{5.1}$$

where $H_{s,\epsilon,n}(\theta) := \frac{\Sigma_s(\theta)}{n} + \frac{\Delta_{s,2}^2}{n^2 \epsilon^2} I$. Here, the multiplication of prior and likelihood densities is not necessarily tractable, and one can use MCMC methods to sample from the posterior distribution effectively. More specifically, Metropolis-Hastings (MH) algorithm explained in chapter 2 can be properly adapted to this setting as the prior is assumed to be available. Algorithm 9 demonstrates an application of MH.

Algorithm 9 MH for the posterior distribution in (5.1) - one iteration

Input: Current sample: θ; privately shared statistic: y, privacy level: ϵ
Output: New sample
Propose $\theta' \sim q(\cdot|\theta)$
Accept the proposal and return θ' with probability

$$\min\left\{1, \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{\eta(\theta')}{\eta(\theta)} \frac{\mathcal{N}(y; \mu_s(\theta'), H_{s,\epsilon,n}(\theta'))}{\mathcal{N}(y; \mu_s(\theta), H_{s,\epsilon,n}(\theta))}\right\}$$

otherwise reject the proposal and return θ.

5.1.2 MH for additive statistic and non-Gaussian noise

In the previous case, Gaussian noise and additive statistic produce approximately normal distribution for the likelihood function of the shared statistic. However, when non-Gaussian noise used as a privacy mechanism, posterior density is not normal distribution anymore, and it is not tractable. On the other hand, additive statistic still allows to use normality assumption for $S_n(X_{1:n})$. As we discussed in the previous chapter, let's define $U = S_n(X_{1:n})$, and joint distribution as

$$\pi(\theta, u|y) \propto \eta(\theta) f_{S_n}(u|\theta) g_{\epsilon,S_n}(y|u). \tag{5.2}$$

Note that, sampling from $\pi(\theta, u|y)$ corresponds to sampling from $p_{\epsilon,S_n}(\theta|y)$ when it is marginalized as

$$\begin{aligned}\int \pi(\theta, u|y)du &= \int \eta(\theta) f_{S_n}(u|\theta) g_{\epsilon,S_n}(y|u)du, \\ &= \eta(\theta) \int f_{S_n}(u|\theta) g_{\epsilon,S_n}(y|u)du = \eta(\theta)p(y|\theta) \quad \text{from equation (4.6)} \\ &= \eta(\theta)p(y|\theta) \propto p_{\epsilon,S_n}(\theta|y).\end{aligned}$$

One of the possible ways of sampling from joint distribution of $\pi(\theta, u|y)$ is using Gibbs sampling, specifically MH-within-Gibbs sampling as conditional densities required to update may not be tractable (See chapter 2). In detail, the method would have two step update mechanism, which is

4.1 Update u using $p(u|\theta, y) \propto f_{S_n}(u|\theta) g_{\epsilon,S_n}(y|u)$,

4.2 Update θ using $p(\theta|u, y) \propto \eta(\theta) f_{S_n}(u|\theta)$.

However, MH-within-Gibbs update mechanism may not be efficient when u, θ and y are highly dependent to each other as it requires more step to escape from the initial phase (Rajaratnam & Sparks, 2015). Fortunately, it is possible to use exact-

approximate MCMC algorithms such as PMMH and MHAAR from chapter 2. Recall, those algorithms mimic the original MH algorithm using sample-based estimators of MH acceptance ratio, and they can overcome the problem of dependency between the variables or non-tractable true-likelihood distributions.

5.1.2.1 Pseudo-marginal MH

Remember that PMMH uses importance sampling to approximate likelihood function when it is not analytically available Andrieu & Roberts (2009). An example of PMMH algorithm targeting the joint distribution in equation (5.2) with intractable likelihood $p_{\epsilon,S_n}(\theta|y)$ is shown in Algorithm 10.

Algorithm 10 PMMH for the posterior distribution in (5.2) - one iteration

Input: Current sample: $(\theta, \hat{Z})$, number of proposals for u: N privately shared statistic y
Output: New sample
Propose $\theta' \sim q(\cdot|\theta)$
Sample $u^{(j)} \sim q_{\theta'}(\cdot)$ for $j = 1, \ldots, N$.
Calculate $\hat{Z}' = \frac{1}{N}\sum_{j=1}^{N} f_{S_n}(u^{(j)}|\theta) g_{\epsilon,n}(y|u^{(j)})/q_{\theta'}(u^{(j)})$ using (4.5).
With probability $\min\left\{1, \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{\eta(\theta')}{\eta(\theta)} \frac{\hat{Z}'}{\hat{Z}}\right\}$, return $(\theta', \hat{Z}')$; otherwise, reject and return $(\theta, \hat{Z})$.

5.1.2.2 MH with Averaged Acceptance Ratios

While PMMH algorithm effectively replaces MH algorithm even when the parts of the posterior distribution is intractable, it sometimes results in sticky Markov chain as we discussed in chapter 2. Additionally, performance of PMMH is closely contingent upon the variability of acceptance ratio (Andrieu & Vihola, 2012; Andrieu et al., 2020; Yıldırım, Andrieu & Doucet, 2018). Fortunately, there is a more recent version of exact-approximate MCMC algorithm which still uses likelihood-free approach but almost completely updates the variables to circumvent the problem of stickiness (Andrieu et al., 2020), and has acceptance ratio whose variance is not increased by n when proposal distribution for θ is scaled. Algorithm 11 targets the posterior distribution in equation (5.2) using MHAAR-Rao-Blackwellised method described in chapter 2,section 2.3.

Algorithm 11 MHAAR-RB for the posterior distribution in (5.2) - one iteration

Input: Current sample: (θ, u); number of proposals for u: N; privately shared statistic: y

Output: New sample

Propose $\theta' \sim q(\cdot|\theta)$

for $j = 1, \ldots, N$ **do**

If $j = 1$ set $u^{(1)} = u$; otherwise sample $u^{(j)} \sim q_{\theta,\theta'}(\cdot)$.

Using (4.5), calculate

$$w_j = \frac{f_{S_n}(u^{(j)}|\theta) g_{\epsilon,n}(y|u^{(j)})}{q_{\theta,\theta'}(u^{(j)})}, \quad w'_j = \frac{f_{S_n}(u^{(j)}|\theta') g_{\epsilon,n}(y|u^{(j)})}{q_{\theta,\theta'}(u^{(j)})}$$

With probability $\min\left\{1, \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{\eta(\theta')}{\eta(\theta)} \frac{\sum_{j=1}^N w'_j}{\sum_{j=1}^N w_j}\right\}$; sample $k \in \{1, \ldots, N\}$ with probability proportional to w'_k and return $(\theta', u^{(k)})$. Otherwise, reject the move, sample $k \in \{1, \ldots, N\}$ with probability proportional to w_k, and return $(\theta, u^{(k)})$.

Note that, this algorithm works properly when the proposal distribution for u satisfies $q_{\theta,\theta'}(u) = q_{\theta',\theta}(u)$ for all θ, θ' and u as explained in Andrieu et al. (2020).

5.1.3 Exact inference based on the true posterior

When the distribution of $S_n(X_{1:n})$ is not available, or the parameters $\mu_S(\theta), \Sigma_S(\theta)$ are not tractable, using previously presented algorithms is not possible to make estimations. In this case, similar to the chapter 4, one can use $x_{1:n}$ instead of $u = S_n(x_{1:n})$. Then, posterior distribution in equation (5.2) can be represented as

$$\pi(\theta, x_{1:n}|y) \propto \eta(\theta) p(x_{1:n}|\theta) p_{\epsilon,S_n}(y|x_{1:n}). \tag{5.3}$$

As a side note, if the distribution of $p(x_{1:n}|\theta)$ is known, augmentation with u in equation (5.3) is still possible.

Moreover, using lower-level representation allows to use distributions that don't depend on θ, hence MHAAR methodology Andrieu et al. (2020) which has advantages over PMMH as mentioned. For this purpose, consider $Z_i \sim \mu(.)$ with the transformation of

$$Z_i \overset{\text{i.i.d}}{\sim} \mu(\cdot) \Rightarrow X_i = \varphi_\theta(Z_i) \overset{\text{i.i.d}}{\sim} \mathcal{P}_\theta, \quad i \geq 1, \tag{5.4}$$

Note that, Z_i's are simply generators of X, and with the proper $\varphi_\theta(.)$, it is possible to consider them as a sequence of random variables from well-defined probability

distribution. With this new representation, joint distribution in equation (5.3) becomes

$$\pi(\theta, z_{1:n}|y) \propto \eta(\theta) h_{\epsilon,S_n}(y|z_{1:n},\theta) \prod_{t=1}^{n} \mu(z_t), \tag{5.5}$$

where $h_{\epsilon,S_n}(y|z_{1:n},\theta) = p_{\epsilon,S_n}(y|x_{1:n})$, with $x_i = \varphi_\theta(z_i)$. More importantly, however, $\pi(\theta, z_{1:n}|y)$ corresponds to our target distribution $p(\theta|y)$ when it is marginalized in a way that

$$\begin{aligned}\int \pi(\theta, z_{1:n}|y) dz_{1:n} &= \eta(\theta) \int h_{\epsilon,S_n}(y|z_{1:n},\theta) \prod_{t=1}^{n} \mu(z_t) dz_{1:n}, \\ &= \eta(\theta) p(y|\theta) \propto p_{\epsilon,S_n}(\theta|y).\end{aligned}$$

Hence, the task is designing a MCMC algorithm targeting $\pi(\theta, z_{1:n}|y)$. Inspiring from Andrieu et al. (2020), Algorithm 12 does this using averaged acceptance ratios.

Algorithm 12 MHAAR for the posterior distribution in (5.5) - one iteration

Input: Current sample: $(\theta, z_{1:n})$, subset size: $m < n$, number of samples for $z_{1:n}$: N, privately shared statistic: y.

Output: New sample

Propose $\theta' \sim q(\cdot|\theta)$

Set $z_{1:n}^{(1)} = z_{1:n}$ and propose $z_{1:n}^{(2)}, \ldots, z_{1:n}^{(N)} \sim \mu(\cdot)$.

Sample $k \in \{1, \ldots, N\}$ with probability proportional to $h_\epsilon(y|z_{1:n}^{(k)}, \theta')$.

Accept $\theta', z_{1:n}^{(k)}$ as the new sample with probability $\min\left\{1, \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{\eta(\theta')}{\eta(\theta)} \frac{\sum_{i=1}^{N} h_\epsilon(y|z_{1:n}^{(i)}, \theta')}{\sum_{i=1}^{N} h_\epsilon(y|z_{1:n}^{(i)}, \theta)}\right\}$; otherwise reject and repeat $(\theta, z_{1:n})$ as the new value.

5.1.4 Exact inference based on the sequential releases

Until now, we focused on inference with summary statistics using notation of S_n, however, one may prefer adding noise sequentially. Recall that, in equation (4.2) we have

$$Y_i = s(X_i) + V, \quad \text{where} \quad V \sim P_{\epsilon,s}.$$

In this setting, it is also possible to use lower-level latent variables Z_i from Algorithm 12 that enables MHAAR methodology. Then, using the same transformation in

equation (5.4), the joint distribution can be adjusted as

$$\pi(\theta, z_{1:n}|y_{1:n}) \propto \eta(\theta) \prod_{t=1}^{n} \mu(z_t) h_\epsilon(y_t|z_t, \theta), \tag{5.6}$$

where $h_\epsilon(y_t|z_t,\theta) = g_{\epsilon,s}(y_t|s(\varphi_\theta(z_t)))$ which corresponds to the $g_{\epsilon,s}(y_t|s(x_t))$ in equation (4.10). As we justified in proposition **??**, one can draw samples from posterior distribution in equation (5.6) with the following acceptance ratio Andrieu et al. (2020);

$$\frac{\prod_{t=1}^{n} \sum_{i=1}^{N} h_\epsilon(y_t|z_t^{(i)}, \theta')}{\prod_{t=1}^{n} \sum_{i=1}^{N} h_\epsilon(y_t|z_t^{(i)}, \theta)}.$$

Note that, only difference between acceptance ratio in previous section is the product over each shared statistics, and this is feasible because each (y_t, z_t) pair is independent. In a similar vein, Algorithm 13 differs from the Algorithm 12 while updating $z_{1:n}$. Since each z_t uniquely determines y_t, one should sample each $z_t^{(k_t)}$ seperately.

Algorithm 13 MHAAR for the posterior distribution in (5.6) - one iteration

Input: Current sample $(\theta, z_{1:n})$, number of samples for $z_{1:n}$: N, privately shared sequence: $y_{1:n}$.

Output: New sample

Propose $\theta' \sim q(\cdot|\theta)$

for $t = 1, \ldots, n$ **do**

  Set $z_t^{(1)} = z_t$ and propose $z_t^{(2)}, \ldots, z_t^{(N)} \sim \mu(\cdot)$.

Calculate the acceptance probability $\alpha = \min\left\{1, \frac{q(\theta|\theta')}{q(\theta'|\theta)} \frac{\eta(\theta')}{\eta(\theta)} \frac{\prod_{t=1}^{n} \sum_{i=1}^{N} h_\epsilon(y_t|z_t^{(i)}, \theta')}{\prod_{t=1}^{n} \sum_{i=1}^{N} h_\epsilon(y_t|z_t^{(i)}, \theta)}\right\}$.

Sample $v \sim \text{Unif}(0, 1)$.

if $v < \alpha$ **then**

  Return $(\theta', z_{1:n} = (z_1^{(k_1)}, \ldots, z_n^{(k_n)}))$, where each $k_t \in \{1, \ldots, N\}$ is sampled with probability proportional to $h_\epsilon(y_t|z_t^{(k_t)}, \theta')$.

else

  Return $(\theta, z_{1:n} = (z_1^{(k_1)}, \ldots, z_n^{(k_n)}))$, where each $k_t \in \{1, \ldots, N\}$ is sampled with probability proportional to $h_\epsilon(y_t|z_t^{(k_t)}, \theta)$.

5.2 Differentially private distributed Bayesian linear regression

In addition to the Bayesian inference on top of the statistic selection presented in chapter 4, one can also focus on inference independent from the statistic selection

using well-known methods such as linear regression. Note that all the work in this section is published at Alparslan, Yıldırım & İlker Birbil (2023).

While differential privacy has become gold-standard for privacy applications lately (Dankar & Emam, 2013; Zhao & Chen, 2022a), there are a few works that directly concerns differential privacy and linear regression. However, some of the ideas from the privacy literature can still be applied on regression domain. As an example, empirical risk minimization (ERM) can be utilized as a way of solving regression problem by minimizing expected value of a loss function over certain dataset (Vapnik, 1991). More importantly, it can be safely combined with differential privacy (Bassily et al., 2014; Kuru, Birbil, Gürbüzbalaban & Yıldırım, 2020; Wang, Chen & Xu, 2019). On the other hand, Bayesian approach through posterior sampling introduces uncertainty and further increases the capability of the prediction model by safely satisfying the privacy guarantees with a careful prior selection (Dimitrakakis et al., 2017; Zhang, Rubinstein & Dimitrakakis, 2016). Hence, more recent studies, including this thesis, employed advanced MCMC methods for Bayesian inference and they are also well-fit for the privatized linear regression problems (Foulds et al., 2016; Heikkilä et al., 2019; Ju, Awan, Gong & Rao, 2022a; Wang et al., 2015; Yıldırım & Ermiş, 2019).

Other than the methods allowing to work on regression problems indirectly, some of the works directly emphasized on differentially private linear regression by proposing variety of methods from objective perturbation to private gradient descent. For instance, Kifer, Smith & Thakurta (2012); Zhang, Zhang, Xiao, Yang & Winslett (2012) proposed adding noise on the objective function of regression problem to satisfy ϵ-DP in equation 3. Bernstein & Sheldon (2019); Wang (2018) suggested sufficient statistics perturbation method that basically add noise on top of the ordinary least squares solution of linear regression. Finally, Liu, Jain, Kong, Oh & Suggala (2023); Varshney, Thakurta & Jain (2022) improved differentially private stochastic gradient algorithms and implemented to the linear regression case which outperformed some of the state-of-art algorithms.

In this thesis, we consider a sufficient statistics perturbation method for the ordinary least squares solution of linear regression as in Wang (2018). Unlikely with that work, we particulary consider a hiearchical model and Bayesian inference, specifically Gibbs sampling (See chapter 2), similar to the one in Bernstein & Sheldon (2019). However, our model also differs from Bernstein & Sheldon (2019) with the summary statistics and resulting hierarchical structure which leads significant performance improvement and remarkable computational advantages.

5.2.1 Notation and privacy mechanism

Linear regression is simply a linear equation that consists of predictors and the response variable which models the dependency on the given data. As a general notation from Myers & Montgomery (1997), one can define a linear regression model as

$$y_i = \beta_0 + \sum_{t=1}^{d} x_{i,t}\beta_t + e_i \quad i = 1, \ldots, n, \tag{5.7}$$

where e_i's are generally chosen as i.i.d. random sequence from a normal distribution (Myers & Montgomery, 1997). Also, ordinary least squares and the maximum likelihood estimator of β is (Myers & Montgomery, 1997)

$$\hat{\beta} = (X^T X)^{-1} X^T y, \tag{5.8}$$

where $X = [x_1^T, \ldots, x_n^T]^T$ and $y = [y_1, \ldots, y_n]^T$. Note that $\{(x_i, y_i) : i = 1, \ldots, n\}$ where $x_i \in \mathcal{X} \subseteq \mathbb{R}^{d \times 1}$ and response variables $y_i \in \mathcal{Y} \subseteq \mathbb{R}$. Normal linear regression model can also be described in matrix notation as

$$y_i = x_i^T \theta + e_i, \quad e_i \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma_y^2), \quad i = 1, \ldots, n,$$

where θ is regression coefficient which corresponds to β in equation (5.7). Additionally, we assume that x_i's identically and independently come from a distribution P_x where it can be assumed as a normal distribution, but we will consider the case where the distribution of x_i is not normal in the following sections. In a matrix notation using the same X and y from the ordinary least squares solution in equation (5.8), one can represent the same model with

$$y = X\theta + e, \quad e \sim \mathcal{N}(0, \sigma_y^2 I).$$

One advantage of the matrix notation is that it becomes easier to observe summary statistics sufficient to obtain ordinary least squares estimation, which are defined as

$$S := X^T X, \quad z := X^T y.$$

Here, there is an important observation regarding the summary statistic.

Proposition 1 *For the normal linear regression model, we have*

$$z | S, \theta, \sigma_y^2 \sim \mathcal{N}(S\theta, S\sigma_y^2).$$

Proof. We know that,

$$\mathbb{E}[z|X,\theta,\sigma_y^2] = \mathbb{E}[X^T X\theta + X^T e] = S\theta, \tag{5.9}$$

$$\mathrm{Cov}(z|X,\theta,\sigma_y^2) = X^T X\sigma_y^2 = S\sigma_y^2, \tag{5.10}$$

Since moments of conditional distribution of z only depends on S,θ,σ_y^2, we can write the density as

$$p(z|X,\theta,\sigma_y^2) = \mathcal{N}(z; S\theta, S\sigma_y^2).$$

Next, define the function $f : \mathbb{R}^{n\times d} \mapsto [0,\infty)$ with $f(X) = p(z|X,\theta,\sigma_y^2)$ and let $\mathcal{C}_{S,\theta,\sigma_y^2} = \{X : X^T X = S\}$, Since the function f is constant over $\mathcal{C}_{S,\theta,\sigma_y^2}$, we can write

$$\begin{aligned} p(z,S) &= \int_{\mathcal{C}_{S,\theta,\sigma_y^2}} f p(S|X)p(X)\mathrm{d}X, \\ &= f\int_{\mathcal{C}_{S,\theta,\sigma_y^2}} p(S|X)p(X)\mathrm{d}X, \\ &= f p(S). \end{aligned}$$

Then, we can arrange the terms and get

$$\begin{aligned} \frac{p(Z,S)}{p(S)} &= f, \\ p(Z|S) &= \mathcal{N}(z; S\theta, S\sigma_y^2). \end{aligned}$$

Another important point is that a well-designed noise injection on top of S and z satisfies differential privacy constraints while inferring $\hat{\beta}$ owing to the composition and post-processing property. Also, working with privatized S and z throughout the MCMC algorithm doesn't increase the cost of privacy (Foulds et al., 2016). Therefore, the privacy setting is

$$\begin{aligned} \hat{S} &= S + \sigma_s M, \\ \hat{z} &= z + \sigma_z v. \end{aligned} \tag{5.11}$$

To satisfy (ϵ,δ)-DP overall, similar to the Bernstein & Sheldon (2019), we consider releasing S and z as a vector by utilizing same noise variance. However, different from Bernstein & Sheldon (2019), we consider using Gauss mechanism as it leads tractable updates in the further stages of Gibss sampling. Then, M is a $d\times d$ symmetric matrix with upper triangular elements coming from distribution $\mathcal{N}(0,1)$ as in Dwork, Talwar, Thakurta & Zhang (2014), and $v \sim \mathcal{N}(0, I_d)$. Additionally, we

prefer calibrating noise variances using the analytic Gauss mechanism from Balle & Wang (2018) as it ensures (ϵ, δ)-DP for wide range of ϵ. Therefore, we have,

$$\sigma_S = \sigma_Z = \Delta_{sz}\sigma(\epsilon, \delta), \tag{5.12}$$

where $\sigma(\epsilon, \delta)$ is numerically calculated using algorithm from Balle & Wang (2018), and Δ_{sz} is combined l-2 sensitivity as

$$\Delta_{sz} = \sqrt{\|X\|^4 + \|X\|^2\|Y\|^2},$$

where $\|X\| = \max_{x\in\mathcal{X}} \|x\|_2$ and $\|Y\| = \max_{y\in\mathcal{Y}} |y|$.

While our model has common features with the model in Bernstein & Sheldon (2019), there are also differences worth noting. In Bernstein & Sheldon (2019), the central limit theorem (CLT) is applied to $\left[S, z, y^T y\right]$, leading to a normality assumption for the whole vector, but this approximation requires fourth order moments as a result of inner products. In contrast, we use the exact conditional distribution $p(z|S, \theta, \sigma^2)$ thanks to Proposition 1, and this distribution is easily identifiable without striving with over demanding calculations. Moreover, we do not require a noisy version $y^T y$, hence have a slight advantage of using less privacy-preserving noise. Additionally, carefully adjusting the prior distributions for the model variables may eliminate the need for normality assumption for a part of sufficient statistics in Bernstein & Sheldon (2019). In this regard, we use the following prior distributions

$$\theta \sim \mathcal{N}(m, C), \quad \sigma_y^2 \sim \mathcal{IG}(a, b), \quad \Sigma_x \sim \mathcal{IW}(\Lambda, \kappa), \tag{5.13}$$

where $\mathcal{IG}$ stands for Inverse-Gamma distribution, and $\mathcal{IW}$ represents Inverse-Wishart distribution. Note that all the distributional parameters $(m, C, a, b, \Lambda, \kappa)$ are predefined model hyper-parameters.

5.2.2 Distributed setting

Recently, many data analytics problems require working on massive datasets where computational burden is increased dramatically. Also, it becomes challenging to store huge amount of information in just one location due to practical and security concerns. Therefore, both data analysts and data holders are more inclined to prefer distributed methods rather centralized solutions (Verbraeken, Wolting, Katzy, Kloppenburg, Verbelen & Rellermeyer, 2020). Although there are more than one

possible way of distributed analysis, working only on the local data is a common practice in these days (Bonawitz, Eichner, Grieskamp, Huba, Ingerman, Ivanov, Kiddon, Konečnỳ, Mazzocchi, McMahan & others, 2019; Dean, Corrado, Monga, Chen, Devin, Mao, Ranzato, Senior, Tucker, Yang, Le & Ng, 2012; Zhang & Lin, 2015) not only for computational advantages but also the fact that data partitioning improves security. In this context, consider the privacy term called "pan-privacy" from Dwork, Naor, Pitassi, Rothblum & Yekhanin (2010) where private data for each individual is stored sequentially, and original data is not available. For this case, pan-privacy can be used to ensure protection of sensitive information when data is partitioned and recorded sequentially. Given all these motivations, we extend our focus of private linear regression to the distributed setting where data is partitioned independently among various data holders. More formally, consider a dataset with $J \geq 1$

$$(X, y) = \{(X_j, y_j); j = 1, \ldots, J\}, \tag{5.14}$$

where number of rows in each X_j is n_j with $n = n_1 + \ldots + n_J$. Similar to the non-distributed setting in (5.11), this time each data node j shares its own summary statistics S_j and z_j in a privacy preserving way as

$$\begin{aligned} \hat{S}_j &= S_j + \sigma_s M_j, \\ \hat{z}_j &= z + \sigma_z v_j, \quad v_j \sim \mathcal{N}(0, I_d). \end{aligned} \tag{5.15}$$

Note that perturbations on the summary statistics are applied on independent parts of the data. Therefore, σ_s, σ_z are all same with non-distributed case presented in equation (5.12). Moreover, $\hat{S}_j$ and $\hat{z}_j$ are statistically more informative than their aggregates $\sum_{j=1}^{J} \hat{S}_j$ and $\sum_{j=1}^{J} \hat{z}_j$ as their sums are not sufficient statistics of noisy versions with respect to θ. Therefore, once the data is partitioned, directly using the noisy versions of the summary statistics with a distributed learning approach is more preferable than aggregating them and continue as a non-distributed setting.

All in all, hierarchical structure depicting the distributed model described above can be seen in figure 5.1

5.2.3 Algorithms for Bayesian inference

After presenting the distributed model and privacy setup, we focus on Bayesian analysis aiming to estimate θ given the noisy summary statistics $\{(\hat{S}_1, \hat{z}_1), \ldots, (\hat{S}_J, \hat{z}_J)\}$.

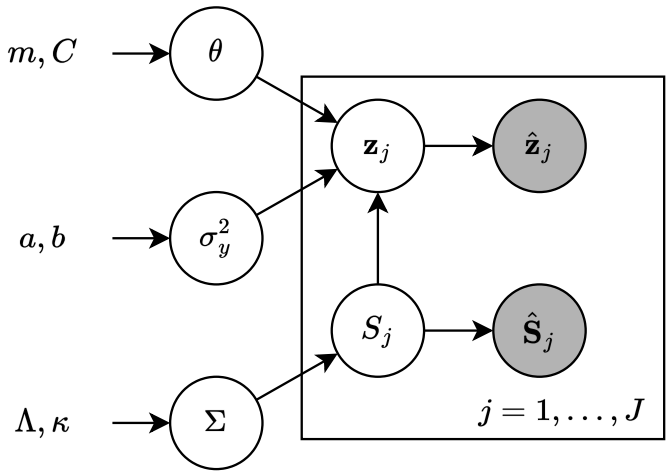

Figure 5.1 Differentially private distributed linear regression model

Roughly, we consider a MCMC methodology that samples from the posterior distribution of the model variables described in hierarachical model given the noisy statistics like we have discussed in section 5.1. For this purpose, distribution of S is vital and closely depends on the distribution of $x \sim P_x$. While specifying P_x as a normal distribution results in exact Wishart distribution for S, it is a strong assumption and may not hold in the real cases. Also, non-Gaussian P_x either requires CLT approximation using up to fourth moments (Bernstein & Sheldon, 2019; Wilson & Ghahramani, 2011), or one may fix S with point estimation of it and use it as a true value which eliminates the $p(S|\Sigma)$ in the sampling chain, and so P_x becomes out of concern. In the following parts, we discuss both perspectives where P_x is normal and unidentified.

5.2.3.1 Normally distributed features

As features are normal, we can consider $P_x = \mathcal{N}(0, \Sigma_x)$ and the prior of $\Sigma_x \sim \mathcal{IW}(\Lambda, \kappa)$. Therefore $S|\Sigma_x \sim \mathcal{W}(\Sigma_x, n)$ Wilson & Ghahramani (2011) where the distributed structure is defined in equation (5.14). Therefore, the posterior distribution of the variables $\theta, \Sigma_x, \sigma_y^2, S_{1:J}, z_{1:J}$ from figure 5.1 can be derived using the independence between each data nodes as

$$p(\theta, \sigma_y^2, \Sigma_x, S, z|\hat{z}, \hat{S}) \propto p(\theta)p(\sigma_y^2)p(\Sigma_x) \prod_{j=1}^{J} p(z_j|\theta, \sigma_y^2, S)p(S_j|\Sigma_x)p(\hat{S}_j|S_j)p(\hat{z}_j|z_j) \tag{5.16}$$

Directly sampling from (5.16) is not possible as it is quite complicated. Hence, one

can design a Gibbs sampler to sample from the posterior distribution (See chapter 2). On the other hand, after some numerical experiments, we realized that posterior parameters θ and $z_{1:J}$ has undesirably high correlation that makes the sampling chain sticky as it was mentioned in chapter 2, section 2.4. Therefore, collapsing (integrating out) the latent variable $z_{1:J}$ from the chain is an option to reduce the effect of correlated variables in the chain (Liu et al., 1994). Therefore, the joint distribution without including $z_{1:J}$ is

$$p(\theta, \sigma_y^2, \Sigma_x, S|\hat{z}, \hat{S}) \propto p(\theta)p(\sigma_y^2)p(\Sigma_x) \prod_{j=1}^{J} p(S_j|\Sigma_x)p(\hat{S}_j|S_j)p(\hat{z}_j|S, \theta, \sigma_y^2), \qquad (5.17)$$

where $p(\hat{z}|S, \theta, \sigma_y^2) = \mathcal{N}(\hat{z}; S\theta, S\sigma_y^2 + \sigma_z^2 I_d)$. Roughly, the reduced model still carries the effect of $z_{1:J}$ but indirectly so that correlation problem that is occurring because of updating one of them conditional to the other one is eliminated.

To design a sampling method for the joint posterior distribution in equation (5.17), one can work with the full conditional distributions, and update the values iteratively as usual in Gibbs sampling. Therefore, firstly, we need to derive full-conditional distributions for the model variables, which are

5.1 $p(\Sigma_x|S_{1:J}, \hat{S}_{1:J}, \hat{z}_{1:J}) \propto p(\Sigma_x) \prod_{j=1}^{J} p(S_j|\Sigma_x)$.

5.2 $p(S_{1:J}|\hat{S}_{1:J}\hat{z}_{1:J}, \Sigma_x, \sigma_y^2, \theta) = \prod_{j=1}^{J} p(S_j|\hat{S}_j\hat{z}_j, \Sigma_x, \sigma_y^2, \theta)$ owing to the factorization, and each factor is $p(S_j|\hat{S}_j\hat{z}_j, \Sigma_x, \sigma_y^2, \theta) \propto p(\hat{z}_j|S, \theta, \sigma_y^2)p(S_j|\Sigma_x)p(\hat{S}_j|S_j)$.

5.3 $p(\theta|\sigma_y^2, \hat{z}_{1:J}, S_{1:J}) \propto p(\theta) \prod_{j=1}^{J} p(\hat{z}_j|S, \theta, \sigma_y^2)$.

5.4 $p(\sigma_y^2|\hat{z}_{1:J}, S, \theta) \propto p(\sigma_y^2) \prod_{j=1}^{J} p(\hat{z}_j|S, \theta, \sigma_y^2)$.

At this point, it is worth mentioning that some of the posterior distributions enjoy closed form representations.

Proposition 2 $p(\Sigma_x|S_{1:J}, \hat{S}_{1:J}, \hat{z}_{1:J}) \sim \mathcal{IW}(\Lambda + \sum_{j=1}^{J} S_j, \kappa + n)$

Proof. We know that

$$\begin{aligned}
p(\Sigma_x|S_{1:J}, \hat{S}_{1:J}, \hat{z}_{1:J}) &\propto p(\Sigma_x) \prod_{j=1}^{J} p(S_j|\Sigma_x) \\
&= \frac{|\Lambda|^{d\kappa/2}}{2^{dk/2}\Gamma_d(\frac{\kappa}{2})} |\Sigma_x|^{-(d+\kappa+1)/2} e^{-\frac{1}{2}\mathrm{tr}(\Lambda\Sigma_x^{-1})} \prod_{j=1}^{J} \frac{|S_j|^{(n_j-d-1)/2} e^{-\frac{1}{2}\mathrm{tr}(\Sigma_x^{-1}S_j)}}{2^{n_j d/2}|\Sigma_x|^{n_j/2}\Gamma_d(n_j/2)} \\
&\propto |\Sigma_x|^{-\frac{n}{2} - \frac{(d+\kappa+1)}{2}} e^{-\frac{1}{2}(\sum \mathrm{tr}(\Sigma_x^{-1}S_j) + \mathrm{tr}(\Lambda\Sigma_x^{-1}))} \\
&\propto |\Sigma_x|^{-\frac{(d+\kappa+n+1)}{2}} e^{-\frac{1}{2}\mathrm{tr}((\sum S_j + \Lambda)\Sigma_x^{-1})}.
\end{aligned}$$

Therefore, we have

$$\Sigma_x|S_{1:J}, S_{1:J}, \hat{z}_{1:J} \sim \mathcal{IW}\left(\Lambda + \sum_{j=1}^{J} S_j, \kappa + n\right).$$

Proposition 3 *$p(\theta|\sigma_y^2, \hat{z}_{1:J}, S_{1:J}) \sim \mathcal{N}(m_p, \Sigma_p)$, where*

$$\Sigma_p^{-1} = \sum_{j=1}^{J} S_j(\sigma_y^2 S_j + \sigma_z^2 I)^{-1} S_j + C^{-1}, \quad m_p = \Sigma_p\left(\sum_{j=1}^{J} S_j(\sigma_y^2 S_j + \sigma_z^2 I)^{-1}\hat{z}_j + C^{-1}m\right).$$

Proof. The posterior of θ is proportional to

$$p(\theta|S_{1:J}, \sigma_y^2, \hat{z}_{1:J}) \propto \mathcal{N}(\theta; m, C)p(\hat{z}_{1:J}|S_{1:J}, \theta, \sigma_y^2).$$

For the second factor, we have

$$\begin{aligned} p(\hat{z}_{1:J}|S_{1:J}, \theta, \sigma_y^2) &\propto \prod_{i=1}^{J} p(\hat{z}_j|S_j, \theta, \sigma_y^2) = \prod_{i=1}^{J} \mathcal{N}\left(\hat{z}_j; S_j\theta, \sigma_y^2 S_j + \sigma_z^2 I\right) \\ &\propto \prod_{i=1}^{J} \exp\left\{-\frac{1}{2}(\hat{z}_j - S_j\theta)^T(\sigma_y^2 S_j + \sigma_z^2 I)^{-1}(\hat{z}_j - S_j\theta)\right\} \\ &\propto \exp\left\{-\frac{1}{2}\left[\theta^T\left(\sum_j S_j(\sigma_y^2 S_j + \sigma_z^2 I)^{-1} S_j\right)\theta - 2\theta^T\left(\sum_j S_j(\sigma_y^2 S_j + \sigma_z^2 I)^{-1}\hat{z}_j\right)\right]\right\}. \end{aligned}$$

Reorganising the terms, we end up with

$$p(\theta|S_{1:J}, \sigma_y^2, \hat{z}_{1:J}) \propto \exp\left\{-\frac{1}{2}\left[\theta^T\Sigma_{\text{post}}^{-1}\theta - 2\theta^T\Sigma_{\text{post}}^{-1}m_{\text{post}}\right]\right\}$$

where $\Sigma_{\text{post}}^{-1} = \sum_j S_j(\sigma_y^2 S_j + \sigma_Z^2 I)^{-1} S_j + C^{-1}$ and $m_{\text{post}} = \Sigma_{\text{post}}[\sum_j S_j(\sigma_y^2 S_j + \sigma_z^2 I)^{-1}\hat{z}_j) + C^{-1}m]$. Therefore,

$$\theta|S_{1:J}, \sigma_y^2, \hat{z}_{1:J} \sim \mathcal{N}(m_{\text{post}}, \Sigma_{\text{post}}).$$

In contrast to the θ and Σ_x, full-conditional distributions of S and σ_y^2 don't yield a tractable distributions. As we mentioned in the chapter 2, this is one of the drawbacks of Gibbs sampling since easy-to-handle conditional distributions may not be available all the time. To alleviate this problem, one can implement one-step MH algorithm targeting to sample from these intractable distributions, which is called MH-within-Gibbs as explained in chapter 2.

Firstly, for the case of S, the factorization enables updating each S_j seperately. Therefore, one can propose $S_j \sim \mathcal{W}(S_j/\alpha, \alpha)$, which has mean S_j and variance de-

termined by α. The main reason behind using Wishart proposal is that it is a general practice in the literature for the covariance matrix update as Wishart distribution is conjugate prior of normal likelihood. Then, the proposed value of S'_j is either accepted or rejected like in classic MH structure. For brevity, algorithm 14 describes one iteration of the sampling procedure.

Algorithm 14 Metropolis-Hastings algorithm for updating S_j - one iteration

Begin with current S_j, α
Propose $S'_j \sim \mathcal{W}(S_j/\alpha, \alpha)$
Accept S'_j and return with probability of

$$\frac{p(S_j|S'_j)}{p(S'_j|S_j)} \frac{p(\hat{z}_j|S'_j, \theta, \sigma_y^2) p(S'_j|\Sigma_x) p(\hat{S}_j|S'_j)}{p(\hat{z}_j|S_j, \theta, \sigma_y^2) p(S_j|\Sigma_x) p(\hat{S}_j|S_j)}$$

else reject the proposed value and return S_j

Fortunately, there is a closed form representation for the first part of acceptance probability, which is

$$\frac{p(S|S')}{p(S'|S)} = \frac{|S|^{(\alpha-d-1)/2}|S|^{\alpha/2}e^{-\mathrm{tr}[aS'^{-1}S]/2}}{|S'|^{(\alpha-d-1)/2}|S'|^{\alpha/2}e^{-\mathrm{tr}[\alpha S^{-1}S']/2}} = \left(\frac{|S|}{|S'|}\right)^{\alpha-(d+1)/2} e^{\alpha(\mathrm{tr}[S^{-1}S']-\mathrm{tr}[S'^{-1}S])/2}.$$

Although Algorithm 14 theoretically converges to the desired posterior distribution of S_j, a key parameter for proposal mechanism, α, may be hard to find for the user. Hence, one can implement adaptive scaling on the hyperparameter α, which makes the chain even more efficient. In this regard, Andrieu & Thoms (2008) proposes a recursive algorithm that simply tries to fix acceptance probability of the method using a gradient-like approach on the parameter α. With the recursive update on α, one can set the acceptance probability to a certain value, and make the initial stages of the chain more useful for the true convergence (Andrieu & Thoms, 2008).

Secondly, we can use simple random-walk with normal distribution (See chapter 2) for designing a MH mechanism to update σ_y^2. Algorithm 15 represents a one-step update procedure in detail.

Hence, the MCMC sampling strategy for distributed linear regression with normally distributed features, which we call `MCMC-normalX`, is available in Algorithm 16.

5.2.3.2 Features with a general distribution

Algorithm 15 Metropolis-Hastings algorithm for updating σ_y^2 - one iteration

Begin with current σ_y^2, and input σ^2_{zprop}
Propose $(\sigma_y^{2'}) \sim \mathcal{N}(\sigma_y^2, \sigma^2_{\text{zprop}})$
Accept $(\sigma_y^{2'})$ and return with probability of

$$\frac{p(\sigma_y^{2'}) \prod_{j=1}^{J} p(\hat{z}_j | S, \theta, \sigma_y^{2'})}{p(\sigma_y^2) \prod_{j=1}^{J} p(\hat{z}_j | S, \theta, \sigma_y^2)}$$

else reject the proposed value and return σ_y^2

Algorithm 16 `MCMC-normalX` - one iteration

Input: Current values of $S_{1:J}$, θ, σ_y^2, Σ_x; observations $\hat{S}_{1:J}$,$\hat{z}_{1:J}$; noise variances σ_s^2, σ_z^2; proposal parameters a, σ_q^2; hyperparameters a, b, κ, Λ, m, C.
Output: New sample of $\Sigma_x, S, \sigma_y^2, \theta$
Sample Σ_x using Proposition 2.
for $j = 1, 2, \dots J$ **do**
  Update S_j using one iteration of Algorithm 14.
Sample θ using Proposition 3.
Update σ_y^2 using one iteration of Algorithm 15.

Normality assumption of the features in the previous model may not be realistic or applicable. Additionally, updating S_j for each j in each iteration may be computationally inefficient. Therefore, one may suggest using an estimator to fix the value of S_j during the whole course of the estimation, which removes the normality assumption. Indeed, after several experiments, we realized that estimating the value of S_j at the beginning, and continue with the same value throughout the iterations results in highly accurate and efficient sampling strategy compared to the cumbersome model, `MCMC-normalX`, especially when the number of nodes J increases.

Clearly, the most important point is the estimation method that replaces the updating step of S_j if it is used as fixed. For this purpose, we can simply consider $\hat{S}_j$ (noisy version of S_j) for estimating S_j. However, the problem is that $\hat{S}_j$ is not necessarily a positive (semi)-definite matrix, which is a must for taking inverse. To tackle this problem, we suggest taking the nearest positive semi-definite matrix in terms of Frobenius norm[1]. Higham (1988) proposes a way to do this which basically uses eigen-decomposition. Consider $\hat{S}_j = EDE^T$ where E is a matrix of eigenvectors and D is a diagonal matrix with eigenvalues in the diagonals. Then, the nearest positive semi-definite matrix according to the Frobenius norm is $\widetilde{S}_j = ED_+E^T$, where D_+ is a diagonal matrix with $D_+(i,i) = \max\{D(i,i), 0\}$. As a side note, taking the

[1]Frobenius norm for matrix $A = (a_{ij})$ can be denoted as $\|A\|_{\text{F}} = (\sum_{i,j} a_{ij}^2)^{1/2}$ Pfeiffer, Kapla & Bura (2021).

Algorithm 17 `MCMC-fixedS` - one iteration

Input: Current values of θ, σ_y^2; observations $\hat{S}_{1:J}$, $\hat{z}_{1:J}$; noise variance σ_z^2, and hyperparameters a, b, m, C.
Output: New sample of σ_y^2,θ.
Use $S_{1:J} = \widetilde{S}_{1:J}$ throughout.
Sample θ using proposition 3.
Update σ_y^2 using one iteration of Algorithm 15.

Algorithm 18 `Bayes-fixedS-fast`

Input: $\hat{S}_{1:J}$, $\hat{z}_{1:J}$; noise variance: σ_z^2; estimate $\tilde{\sigma}_y^2$ of σ_y^2; hyperparameters: m, C.
Output: Estimate $\hat{\theta}$.
for $j = 1, 2, \dots J$ **do**
  Calculate the estimate $\widetilde{S}_j$ for S_j using $\hat{S}_j$.
  Calculate $\Sigma_j = \widetilde{S}_j(\tilde{\sigma}_y^2\widetilde{S}_j + \sigma_z^2 I)^{-1}\widetilde{S}_j$.
  Calculate $m_j = \widetilde{S}_j(\tilde{\sigma}_y^2\widetilde{S}_j + \sigma_z^2 I)^{-1}\hat{z}_j$.
Calculate $\Sigma_{\text{post}}^{-1} = \sum_{j=1}^{J}\Sigma_j + C^{-1}$, $m_{\text{post}} = \Sigma_{\text{post}}\left(C^{-1}m + \sum_{j=1}^{J} m_j\right)$.
return $\hat{\theta} = \Sigma_{\text{post}}^{-1} m_{\text{post}}$

nearest positve semi-definite matrix according to the Frobenius norm as an estimation of S_j is equivalent of taking the maximum likelihood estimation of S_j since the likelihood $p(\hat{S}_j|S_j)$ is normally distributed. Hence, a detailed representation of this methodology, which we call `MCMC-fixedS`, is available in Algorithm 17.

Algorithm 17 eliminates the need of normality assumption by removing the update of S as the distribution of P_x only concerns the distribution of $p(S|\Sigma_x)$. Additionally, Algorithm 17 is computationally more efficient than the Algorithm 16 as one of the parts requiring MH sampling is already removed from the sampling strategy. However, one can go further and take out the other MH part from the chain to make it even faster. After several numerical experiments, we realized that replacing σ_y^2 with a crude estimator $\tilde{\sigma}_y^2 = \|\mathcal{Y}\|/3$ results in higher efficiency. This method, `Bayes-fixed S-fast`, still utilizes Bayesian perspective while estimating θ, but doesn't update S. Also, it is faster than the other algorithms. In fact, `Bayes-fixedS-fast` is not even a MCMC algorithm, instead it is an one-step calculation of $\hat{\theta}$ given the estimations of S_j, σ_y^2 and hyperparameters.

5.2.4 Variants of the proposed methods

While it is possible to remove key part regarding normality assumption from the chain to deal with non-normal features, one can also use averaging to obtain approximately normal data. Additionally, linear regression models also include bias term, or so called intercept parameter. Proposed algorithms mostly ignore the existence of the intercept term whereas including it in the model may affect the distributional assumption negatively. As a result, we resort this section for the discussion possible variants of the models. On the other hand, we consider these models as hypothetical and possibly useful for future studies while we don't include them in the evaluation and numerical experiments.

5.2.4.1 Another way of dealing with non-normality

When x_i, $i = 1, \ldots, n$ are not normal, another approach is based on modifying the data to such that the rows of the modified feature matrix, called X_{av}, are averages of $k > 1$ original features in X, and thus approximately normal, by the CLT. Specifically, let n be divisible by k so that $m = n/k$ is an integer. Consider the $m \times n$ matrix

$$A = \frac{1}{\sqrt{k}} \begin{bmatrix} 1_{1\times k} & 0_{1\times k} & \ldots & 0_{1\times k} \\ 0_{1\times k} & 1_{1\times k} & \ldots & 0_{1\times k} \\ \vdots & \vdots & \ddots & \vdots \\ 0_{1\times k} & 0_{1\times k} & \ldots & 1_{1\times k} \end{bmatrix}_{m\times n},$$

Then the matrix $X_{\text{av}} = AX$ corresponds to constructing a shorter $m \times d$ matrix whose i'th column is the average of the rows $(i-1)k+1, \ldots, ik$ of X (scaled by $1/\sqrt{k}$ to preserve the norm). When k is large enough, we can make normality assumptions for the rows of X_{av}. Further, we consider

$$y_{\text{av}} := Ay = X_{\text{av}}\theta + Ae,$$

whose mean is $X_{\text{av}}\theta$ and covariance $AA^T\sigma_y^2$. But, we have $AA^T = I_m$, so the covariance is $\sigma_y^2 I_m$. Therefore, the same hierarchical model in Figure 5.1 can be used for X', y' with their respective summary statistics

$$z_{\text{av}} = (X_{\text{av}})^T y_{\text{av}}, \quad S_{\text{av}} = (X_{\text{av}})^T X_{\text{av}},$$

as well as the noisy versions of those summary statistics to provide a given level of privacy. Note that S_{av} and z_{av} have the same sensitivities as S and z, hence the same noise variances are needed for privacy. However, there is less information in

S_{av} and z_{av} due to averaging.

5.2.4.2 What happens when we include intercept?

Intercept parameter corresponds to appending x_i with a 1 from the left. Then, $S = X^T X$ becomes $S_0 = \begin{bmatrix} n & n\bar{x}^T \\ n\bar{x} & S \end{bmatrix}$, where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$. Also, note that $S = (n-1)\widehat{\Sigma}_x + n\bar{x}\bar{x}^T$ where $\widehat{\Sigma}_x$ is the sample covariance. Under the normality assumption for x_i's, $\bar{x} \sim \mathcal{N}(m, \Sigma_x/n)$ and $(n-1)\widehat{\Sigma}_x \sim \mathcal{W}(n-1, \Sigma_x)$ are independent and have known distributions. Therefore, we can write a model that includes $b = \bar{x}$, $\hat{\Sigma_x}$, and S_0 where S_0 replaces S in the standard model. More specifically, we have the following hierarchical model:

$$\theta \sim \mathcal{N}(m, C), \quad \Sigma_x \sim \mathcal{IW}(\Lambda, \kappa), \quad \hat{\Sigma_x}|\Sigma_x \sim \mathcal{W}(n-1, \Sigma_x), \quad b|\Sigma_x \sim \mathcal{N}(\mu, \Sigma_x/n),$$
$$z|\theta, \Sigma_y^2, \hat{\Sigma}, b \sim \mathcal{N}(S_0\theta, S_0\sigma_y^2), \quad \hat{S}|\hat{\Sigma}, b = \mathcal{N}(S_0, \sigma_s^2 I), \quad \hat{z}|z = \mathcal{N}(z, \sigma_z^2 I)$$

with

$$S_0 = \begin{bmatrix} n & nb^T \\ nb & (n-1)\hat{\Sigma} + nbb^T \end{bmatrix}.$$

6. Experiments for the inference with statistic selection

In this chapter, we would like to demonstrate effectiveness of our statistic selection method using numerical simulations. While doing so, we emphasize that statistic selection methodology with Fisher information presented in chapter 4 actually suggests a strategy that leads less error when it is combined with the inference method mentioned in chapter 5. To this end, each different scenario in statistic selection topic is analyzed separately in the following sections.

One can easily obtain the performance of a Bayes estimator using mean squared error (MSE) as higher MSE means more dispersion from the desired results in terms of variance and bias. More specifically, given $\hat{\theta}(Y) = \mathbb{E}(\theta|Y)$, and θ^* is the true value of sample statistic, MSE for Bayes estimator is

$$\begin{aligned} \text{MSE} &= \mathbb{E}_Y[(\hat{\theta}(Y) - \theta^*)^2], \\ &= \text{Var}(\hat{\theta}(Y)) + \text{Bias}^2(\hat{\theta}(Y)). \end{aligned}$$

Note that, MSE definition above requires an integral calculation over the distribution of Y. Remembering the equation (2.3) in chapter 2, it is approximately equals to

$$\mathbb{E}_Y[(\hat{\theta}(Y) - \theta^*)^2] \approx \frac{1}{M} \sum_{i=1}^{M} ((\hat{\theta}(Y^{(i)}) - \theta^*)^2),$$

where M independent samples of $Y^{(i)}$ are drawn from the distribution $p(Y|\theta^*)$. All of the results and discussions in this section are published at Alparslan & Yıldırım (2022).

6.1 Comparison of additive statistic with the Gauss mechanism

This part concerns when the summary statistic is designed as additive statistic and the noise perturbation is the Gaussian mechanism. In this section, we focus on the example in 4 in chapter 4 where $P_\theta = \mathcal{N}(0,\theta)$, $X \in [-A, A]$, and $s(X) = |x|^a$. As a side note, the following comparison is also true for other possible cases, but variance of a normal distribution is particularly is a well-known problem in statistical analysis, which justifies our reasoning behind focusing on this particular case.

To make the comparison easy-to-follow, consider $a \in \{1,2\}$ as we used in example 4, and remember that $s(X) = |x|$ was more informative than its counterpart when there is DP noise or vice versa. Following that, we carry out the comparison between the MSE values of estimation by Algorithm 9 when $a = 1$ and $a = 2$ respectively to obtain the harmony between statistic selection methods and Bayesian estimation. During the simulation runs, we take $A = 10$ to wash out the effect of boundedness on the sensitivity calculations, $n = 100$, $\theta^* = 2$, $M = 10^3$. More importantly, we use an uninformative prior on θ and random-walk for the proposal, which eliminates the parts other than the likelihoods in the acceptance probability calculation. After running the MH algorithm $K = 10^5$ iterations and averaging the output samples to obtain a single estimation, we obtain the values in figure 6.1.

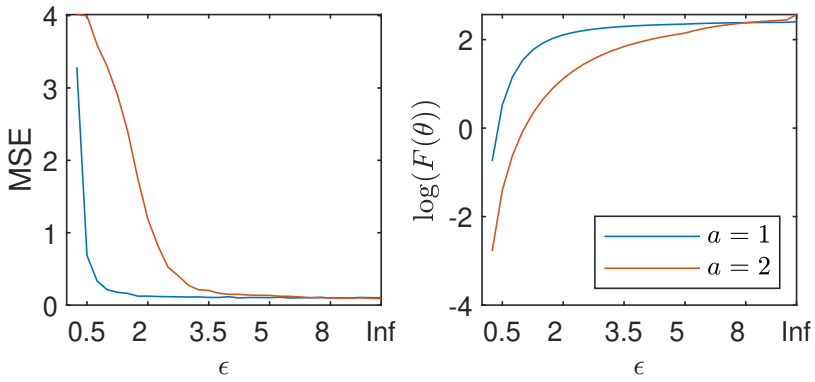

Figure 6.1 MSE and (Logarithm of) $F(\theta)$ for different moments when there is Gaussian noise.

One can easily deduce from the figure that MSE values of the estimations agrees with the suggestion of the statistic selection methodology for all privacy levels. Realize that, $F(\theta)$ for $s(X) = |x|$ is always highe, and accordingly MSE of the estimation when $a = 1$ is always lower than the counterpart. Hence, Fisher information is actually a prominent and neat way of selecting a summary statistic for differentially private Bayesian analysis.

6.2 Comparison of additive statistic with the Laplace mechanism

This time, additive statistic is shared with Laplace noise rather than the Gaussian noise while implementing the same example in the previous section with the same structure ($P_\theta = \mathcal{N}(0,\theta)$, $X \in [-A, A]$, $s(X) = |x|^a$). Also, remember that, $s(X) = |x|$ is more informative when $\epsilon = 1$.

Target of the comparison is still same, i.e. whether the more informative statistic according to Fisher information also results in better performance when it is utilized by a MCMC strategy or not. Differing from the previous section, one can use either Algorithm 10 or Algorithm 11 as an inference tool while we prefer conducting this comparison with Algorithm 10 due to its simplicity. However, in the following part we compare both of these algorithms in terms of their mixing properties in detail, and as expected Algorithm 11 beats over. All in all, after using the same parameters and averaging over $M = 10^2$ noisy observations, results are obtained in figure 6.2.

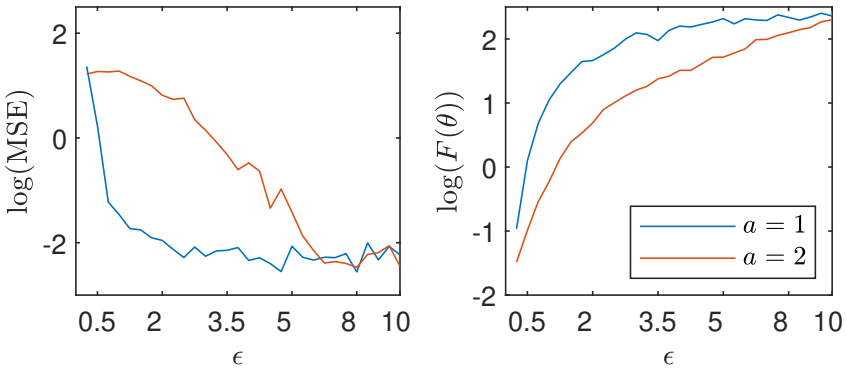

Figure 6.2 MSE (left) and $F(\theta)$ (right) for $s(x) = |x|$ (blue) and $s(x) = x^2$ (red), under Laplace mechanism. MSE is calculated from the samples obtained from Algorithm 10.

As in the case where additive statistic and Gaussian noise employed, figure 6.2 also reveals that Fisher information actually suggests the true statistic to share with noise when more effective Bayesian analysis is desired to be carried out for the combination of additive statistic and Laplace mechanism.

6.2.1 Comparison of Algorithms 10 and 11 in terms of mixing

As it was presented in chapter section 5.1 in chapter 5, it is possible to make inference with both Algorithms 10 and 11 for the case of additive statistic and non-Gaussian (Laplace) noise. Hence, a discussion on the comparison of the performances of these two algorithms for the same problem is noteworthy.

There are several methods to asses the performance of a MCMC algorithm. For instance, one can check the convergence status of a chain for a certain amount of iterations (Robert & Casella, 2004). Various diagnostic methods have been proposed for this purpose such as Kolmogorov-Smirnov test (Lehmann & D'Abrera, 1998), or naive visual checks on the plots visualizing the evolution of the samples according to iterations. However, other than these diagnostic tools, one can also check integrated auto-correlation (IAC) time, which is the asymptotic variance of an average of samples generated by the MCMC algorithm relative to that of the average of i.i.d samples from the target distribution. Roughly, IAC is an effective measure of mixing behavior of the chain, and the larger IAC means the more samples are needed to converge and truly represent the target distribution (Foreman-Mackey, Hogg, Lang & Goodman, 2013). Hence, smaller IAC time is favorable.

For the comparison in previous section, we have a winner statistic, $s(x) = |x|$. Therefore, while comparing IAC times of these algorithms we focus on the case where $a = 1$. Also, due to computational complexity, we just compare them when DP parameter $\epsilon = 5$. In terms of algorithmic specifications, both of them still uses random-walk proposal and uninformative prior. Additionally, the importance sampling parameter for Algorithm 10 is taken as $q_\theta(u) = f_{S_n}(u|\theta)$, whereas the symmetric proposal distribution of u in Algorithm 11 is taken as $q_{\theta,\theta'}(u) = f_{S_n}(u|(\theta+\theta')/2)$. Table 6.1 shows the values of IAC times of the algrorithms compared to various N where it denotes the number of particles per iteration.

Table 6.1 IAC values of Algorithms 10 and 11 versus N

N	Algorithm 10	Algorithm 11
2	44.03	17.99
5	28.19	17.10
10	21.11	16.13
20	18.16	15.44
50	15.32	13.78
100	16.42	15.86

It is clear that Algorithm 11 performs better for all values of N, and this difference is a result of sophisticated structure of MHAAR. More specifically, one can state that completely refreshing all variables improves the mixing property of the chain.

6.3 Comparison of non-additive statistic

In the previous section, we changed the noise mechanism while keeping the summary statistic same. However, in this part, the main focus is shifted to the non-additive statistics as in chapter 4 and chapter 5. Specifically, we consider following statistics for the inference;

$$S_n(X_{1:n}) = \max\{s(X_i); i = 1, \ldots, n\}, \tag{6.1}$$

$$S_n(X_{1:n}) = \text{median}\{s(X_i); i = 1, \ldots, n\} \tag{6.2}$$

Remember that, as it was discussed in chapter 3, adding noise on top of the non-additive statistics based on global sensitivity is ineffective since global sensitivity is indenpendent from data size n and it may easily get out of control which may result in excessively large noise on the summary statistic. Hence, we consider using smooth sensitivity derived in definition 7 by setting $A_s = \max_{x\in\mathcal{X}} s(x)$, and letting $\min_{x\in\mathcal{X}} s(x) = 0$. Therefore, the smooth sensitivity for the maximum in (6.1) is given by,

$$\Delta^{smooth}_{\max,\beta}(x_{1:n}) = \max\{e^{-k\beta} b_k; k = 0, \ldots, n\},$$

with $b_k = \max\{A_s - s_{n-k}, s_n - s_{n-k-1}\}$. For the median in in (6.2), the smooth sensitivity is

$$\Delta^{smooth}_{\text{med},\beta}(x_{1:n}) = \max\{e^{-k\beta} b_k; k = 0, \ldots, n\}$$

with $b_k = \max\{s_{m+i} - s_{m+i-k-1}; i = 0, \ldots, k+1\}$. Note that, in both calculations, $s_1, \ldots, s_n$ are the sorted values of $s(x_1), \ldots, s(x_n)$ so that $0 \leq s_1 \leq s_2 \leq \ldots \leq s_n \leq A_s$.

For the sake of completeness, this implementation is also based on the example 4 where $P_\theta = \mathcal{N}(0, \theta)$, $X \in [-A, A]$ and $s(x) = \{|x|, x^2\}$, but with a small difference of just using $s(x) = |x|$ as it is the winner statistic up-until now. Additionally, we prefer using Laplace mechanism for the comparison. Since the normality assumption on the distribution of $S_n(X_{1:n})$ is not feasible anymore, Gaussian noise doesn't lead us to the closed form. Therefore, we stick up with the base definition of DP using Laplace noise for this case.

For the experiment, we use Algorithm 12, and we take the parameters as $n = 100, \delta = 1/n^2, M = 100$, and $N = 10^4$ for latent particles. Differing from the previous comparisons, we only focus on $\epsilon = 5$ because of the high computational cost of the smooth sensitivity calculations. At the end, table 6.2 demonstrates the MSE values according to various θ values for these non-additive statistics. These results clearly shows

that median statistic is way more efficient than the maximum statistic in terms of MSE.

Table 6.2 MSE for median and maximum statistics

$S_n(X_{1:n})$	$\theta = 0.5$	$\theta = 1$	$\theta = 2$
median	0.027	0.061	0.391
max	10.80	15.57	22.64

Now, comparison of the Fisher information is required to justify our statistic selection method. Figure 6.3 concludes the discussion by admitting the obvious superiority of median statistic also in terms of the Fisher information calculated by using Algorithm 7.

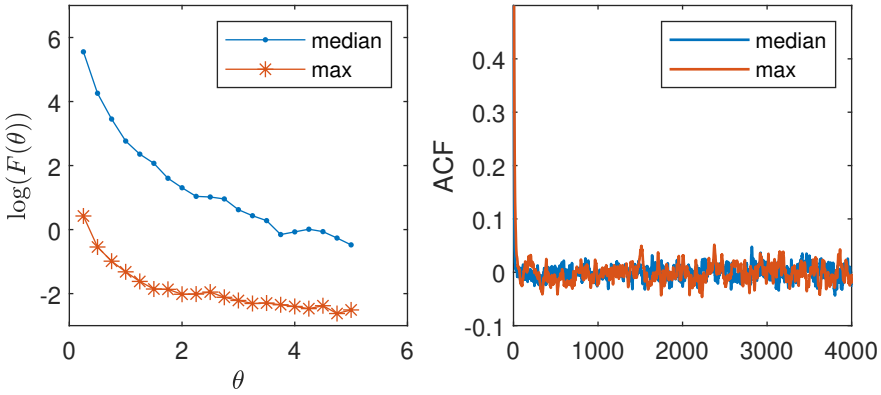

Figure 6.3 Left: $F(\theta)$ for median (blue) and maximum (red) of $s(x) = |x|$. Right: Autocorrelation function (ACF) for Algorithm 12 for median (blue) and maximum (red) at $\theta = 2$. Privacy parameters are $(\epsilon, \delta) = (5, 1/n^2)$.

In addition to the $F(\theta)$, right part of the figure 6.3 presents auto-correlation function (ACF) calculated at $\theta = 2$ and averaged over 5 runs for each noisy observation. ACF is also an useful measure of the mixing behavior of the MCMC algorithm. After plotting the values according to the iteration number, one can obtain oscillations around 0 for ACF, which means that samples from the chain have smaller correlation among them, so they convergence faster to the desired posterior distribution. Therefore, MSE calculations are reliable.

6.4 Comparison of sequential release

Following the same outline, this time our focus is the case where noisy observations are released sequentially. As it was discussed in section 5, Algorithm **??** is designed to sample θ from the posterior distribution in equation (5.6).

Experimental structure in this section is almost same with the one including additive statistic with Laplace mechanism. Namely, we employ the same structure in example 4, which is $P_\theta = \mathcal{N}(0, \theta)$, $X \in [-A, A]$, $s(x) = \{|x|, x^2\}$, and we Laplace mechanism for the privacy protection. Additionally, some of the hyper-parameters for the experiments are $n = 100, M = 10^4$, and $N = 10^4$ for number of latent particles.

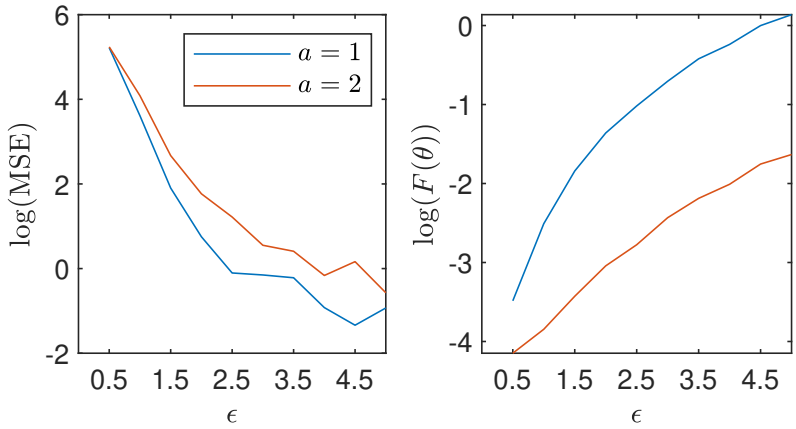

Figure 6.4 MSE (left) and $F(\theta)$ (right) for $s(x) = |x|$ (blue) and $s(x) = x^2$ (red), under Laplace mechanism using sequential release. MSE is calculated from the samples obtained from Algorithm 13.

Figure 6.4 demonstrates the comparison of the statistics $s(x) = |x|, x^2$, and the results are aligned with the previous observations. Namely, the one reveals more information about the posterior distribution results in terms of Fisher information also results in less MSE during the Bayesian inference.

6.5 Comparison based on the initial data

Up until now, all the experiments initially assume that one statistic is more informative than the other and measures their efficiency in terms of the MSE based on this assumption. However, there may be a case where one statistic does not clearly outperform the other statistics according to the Fisher information which may result in confusion while using the statistic selection scheme. In fact, one of the examples described before (Example 5) resembles this situation for the width of an

uniform distribution. One may think that it is possible to use prior information on θ to measure informativeness of the statistics. However, this is only the case when informative prior is utilized for the inference as the name suggests. On the other hand, throughout the numerical comparisons, we only consider uninformative priors to simplify estimation procedure. Hence, we need more sophisticated approach for this cases to enhance capabilities of our statistic selection method.

The method accounting initial data simply starts by splitting the whole sensitive data into two parts as $X_{1:n_0}$ and $X_{n_0+1:n}$, where $n_0 < n$ is a small fraction of n. The first part is used for getting initial information about θ by determining baseline informativeness of the statistics using Fisher information. After deciding the best statistic using the initial chunk, rest of the data is used for learning the value of θ based on the initial information. More specifically, assuming the summary statistic is additive (non-additive statistic is also applicable with small change) and ϵ-DP, the method has following steps;

- Based on the arbitrarily chosen s_0 (for instance $s_0(x) = |x|$), calculate noisy observation using $X_{1:n_0}$

$$Y_0 = \frac{1}{n_0} \sum_{i=1}^{n_0} s_0(X_i) + V_0,$$

  where V_0 is the privacy-preserving noise arranged to satisfy ϵ-DP for the initial chunk of data.

- Using a non-informative prior distribution $\eta(\theta)$, we obtain samples $\theta_0^{(1)}, \ldots, \theta_0^{(M)}$ from the posterior distribution of θ conditional on $Y_0 = y_0$.

- We use those samples for statistic selection: For each candidate statistic s, we calculate its score as

$$\frac{1}{M} \sum_{i=1}^{M} F_s^{-1}(\theta_0^{(i)}),$$

  where $F_s(\theta)$ is the Fisher information for when s is used. The above average is the approximation to $\int F^{-1}(\theta) p(\theta|y_0) d\theta$.

- Among the candidates, the statistic with the lowest score is selected for sharing the remaining part of the data, $X_{n0+1:n}$. Call the selected statistic s^*. The remaining data is shared as

$$Y_1 = \frac{1}{n - n_0} \sum_{i=n_0+1}^{n} s^*(X_i) + V_1,$$

  where V_1 is the privacy-preserving noise arranged to satisfy ϵ-DP for the re-

maining part of the data. Note that, since $X_{1:n_0}$ and $X_{n_0+1:n}$ are disjoint, performing ϵ-DP operations on each part separately satisfies ϵ-DP overall.

- Based on $Y_0 = y_0$ and $Y_1 = y_1$ (and the respective statistics that are used to generate them), we perform Bayesian estimation of θ with the prior distribution $\eta(\theta)$.

Since s^* is obtained with a careful consideration on the prior information about the effectiveness of the statistics, estimation of θ using s^* is expected to be more accurate than just using arbitrary s_0 for whole inference process.

We compare this method with the no statistic selection, which is only utilizing s_0, for the setting explained in example 5. Namely, $P_\theta = \text{Unif}(-\theta, \theta)$, $X \in [-A, A]$, and the aim is inferring θ. For the simulation purpose, we consider $n = 100$, $n_0 = 10$, and $s_0(x) = |x|$. In terms of statistic selection, our purpose is deciding on most informative one among the set $s(x) = \{|x|^a;\, a = 0.1, 0.2, \ldots, 2\}$. For the true value of $\theta^* = 0.1, 0.2, \ldots, 1$, we compare MSE values with the estimated theta valuse based on the y_1.

All in all, figure 6.5 shows the merits of the statistic selection methodology with 15000 runs for each θ^*. As it is obvious in the left-part of the figure, making an estimation based on the carefully decided statistic increases efficiency in terms of MSE.

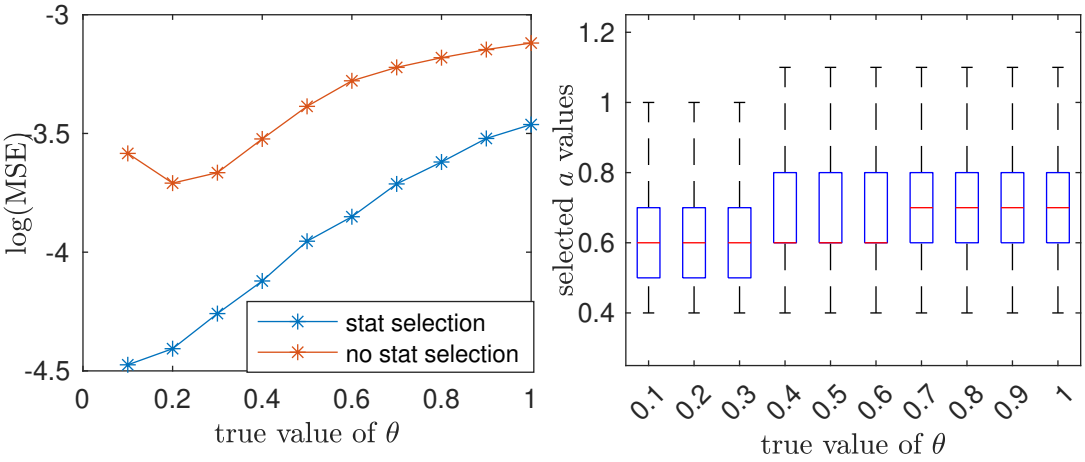

Figure 6.5 Left: MSE values with and without statistic selection using initial data. Right: Box-plots (outliers removed) of selected a values when statistic selection is performed.

Additionally, in the right part of the figure, one can see that range of chosen statistic dynamically changes according to the true value of θ. Hence, utilizing arbitrarily chosen statistic is not a wise choice as it may not be the best choice given the simulation setting.

7. Experiments for the private linear regression

We resort this chapter for the numerical experiments of the methods presented in section 5.2 in chapter 5. While the focus was testing the effectiveness of the statistic selection method in the previous experimental section, this time the concern is both efficiency and the effectiveness of the inference methods in the linear regression setting described previously. For this purpose, comparison with the existing state-of-art methods is possibly the most prominent way to measure the performance. Up to our knowledge, two strategies from the literature outshine others, `adaSSP` from Wang (2018) and `MCMC-B&S` from Bernstein & Sheldon (2019). In fact, it was mentioned that Bayesian inference algorithms proposed previously for the distributed and private linear regression problem have similarities with these two methods. However, it is crucially important to mention that `adaSSP` and `MCMC-B&S` are not completely applicable to the distributed setting as they are not designed for that purpose. Therefore, they need implementation-wise extensions for the fair comparison with our methods which can be found in detail in the following sections. As a metric for comparison, mean squared error (MSE) is again a principal method but this time we solely focus on the prediction performances of the methods as the prediction is one of the main use-cases for the regression methods. For measuring this prediction performance, we use $\mathbb{E}[\hat{y}(x_{\text{test}}) - y_{\text{test}}]^2$ where the whole data is splitted as test and train accordingly. For the Bayesian methods, $\hat{y}(x_{\text{test}})$ is the posterior predictive expectation of y_{test} at x_{test}. For `adaSSP`, we simply take $\hat{y}(x_{\text{test}}) = x_{\text{test}}^T \hat{\theta}$. In particular, this comparison is carried out for both artificial and existing data to get more realistic evaluation at the end. Additionally, along with the MSE, we also evaluate calibration of the posterior predictive distribution using confidence intervals and maximum mean discrepancy (MMD) following Bernstein & Sheldon (2019), which is

$$\text{MMD}^2(P^1, P^2) = \frac{1}{n(n-1)} \sum_{i \neq j}^{n} (k(p_i^1, p_j^1) + k(p_i^2, p_j^2) - k(p_i^1, p_j^2) - k(p_j^1, p_i^2)),$$

where $(p^1, p^2) \sim P^1, P^2$, and $k(x,y) = \frac{1}{2\pi} e^{-\frac{x^2+y^2}{2}}$ -i.e. standard normal kernel. In short, higher value of MMD means that P^1 and P^2 are more likely to be different distributions from each other. Since diverting from the non-private distribution is undesirable, one can use MMD to measure the difference between private and non-private posterior distributions. These results are published at Alparslan et al. (2023).

7.1 Extensions on state-of-art methods

7.1.1 Distributed `adaSSP`

Originially, `adaSSP` is designed for the single data holder and is not applicable to the multi-party data sharing case Wang (2018). The algorithm simply returns estimation of the regression coefficients $\hat{\theta}$ as

$$\hat{\theta} = (\hat{S} + \lambda I)^{-1} \hat{z}. \tag{7.1}$$

While the $\hat{S}$ and $\hat{z}$ are the private versions of $X^T X$ and $X^T y$ respectively, regularization coefficient of λ is also included and privatized to improve the performance of the estimation. As a result of releasing three statistics with noise, due to the composition principle in definition 14, each process utilizes $(\epsilon/3, \delta/3)$-DP. Note that releasing regularization coefficient with noise requires non-standard operations by employing eigenvalue of S. Hence, implementing `adaSSP` on a distributed data setting may be intricate.

Although extending this method for the distributed problem is not straightforward, one can easily replace the estimation in (7.1) with the aggregates as

$$\hat{\theta} = \left(\sum_{j=1}^{J} \hat{S}_j + I \sum_{j=1}^{J} \lambda_j \right)^{-1} \left(\sum_{j=1}^{J} \hat{z}_j \right). \tag{7.2}$$

Here, $\hat{S}_j$, $\hat{z}_j$ and λ_j are calculated in data node j separately from the other nodes. The estimation procedure in (7.2) does not properly account for the optimal (regu-

larised) least squares solution but approximates it.

7.1.2 Distributed and multidimensional `MCMC B&S`

Like `adaSSP`, `MCMC B&S` in Bernstein & Sheldon (2019) also considers $J = 1$, and the vector $ss = [\text{vec}(S), z = X^T y, u = y^T y]$ is perturbed with privacy-preserving noise to generate the observations of the model. For $J \geq 1$, the following natural extension can be considered for generating perturbed observations $\hat{ss} = [\text{vec}(\hat{S}_j), \hat{z}_j, \hat{u}_j]$ along with

$$\hat{S}_j = S_j + \sigma_{dp} M_j, \quad \hat{z}_j = z_j + v_j, \quad v_j \sim \mathcal{N}(0, \sigma_{dp}^2 I_d), \quad \hat{u}_j = u_j + w_j, \quad w_j \sim \mathcal{N}(0, \sigma_{dp}^2),$$

where $\sigma_{dp} = \sigma(\epsilon, \delta)\Delta_{ss}$ with $\Delta_{ss} = \sqrt{\|\mathcal{X}\|^4 + \|\mathcal{X}\|^2\|\mathcal{Y}\|^2 + \|\mathcal{Y}\|^4}$.

For the sake of completeness, we provide the further specifics of the model: We take $(\theta, \sigma_y^2) \sim \mathcal{NIG}(a_0, b_0, m, \Lambda_0)$ where $\Lambda_0 = C^{-1}$ and $P_x = \mathcal{N}(0, \Sigma_x)$ with $\Sigma_x \sim \mathcal{IW}(\Lambda, \kappa)$.

During the comparisons, we set $a_0, b_0, m, C, \Lambda, \kappa$ to the same values for both this model and our proposed model that assumes normally distributed features, i.e. $P_x = \mathcal{N}(0, \Sigma_x)$. Then, we apply an extension of (Bernstein & Sheldon, 2019, Algorithm 1) suited to those observations. One iteration of that algorithm includes the following steps in order:

- Calculate the $D \times 1$ mean vector and $D \times D$ covariance matrix

  $$\mu_{ss} = \mathbb{E}[ss], \quad \Sigma_{ss} = \text{Cov}[ss].$$

  This step requires the fourth moments of $\mathcal{N}(0, \Sigma_x)$.

- Sample $ss_j \sim \mathcal{N}(\mu^{(j)}_{\text{post},ss}, \Sigma^{(j)}_{\text{post},ss})$ with

  $$\Sigma^{(j)}_{\text{post},ss} = (n_j \Sigma_{ss}(\theta)^{-1} + (1/\sigma_{dp}^2) I)^{-1}, \quad \text{and} \quad \mu^{(j)}_{\text{post},ss} = \Sigma^{(j)}_{\text{post},ss}(\Sigma_{ss}(\theta)^{-1}\mu_{ss} + \hat{ss}_j/\sigma_{dp}^2).$$

- Sample $\Sigma_x \sim \mathcal{IW}\left(\Lambda + \sum_{j=1}^J S_j, n + \kappa\right)$.

- Sample $(\theta, \sigma_y^2) \sim \mathcal{NIG}(a_n, b_n, m_n, \Lambda_n)$ by sampling $\sigma_y^2 \sim \mathcal{IG}(a_n, b_n)$, followed by sampling $\theta \sim \mathcal{N}(m_n, \sigma_y^2 \Lambda_n^{-1})$ with $a_n = a_0 + n/2$, $b_n = 0.5u + m^T C^{-1} m -$

$m_n^T \Lambda_n m_n$, and

$$\Lambda_n = \Lambda_0 + \sum_{j=1}^{J} S_j, \quad m_n = \Lambda_n^{-1} \left(\sum_{j=1}^{J} z_j + \Lambda_0 m \right).$$

Note that the first step requires utilizing moments of $\mathcal{N}(0, \Sigma_x)$ upto fourth degree, and calculating these values becomes complicated when the data dimension increases. Indeed, original method in Bernstein & Sheldon (2019) considers only one dimensional data case during the numerical experiments. To further extend this algorithm and enable higher dimensions for the fair comparison, we consider moment calculation method in Triantafyllopoulos (2002).

7.2 Experiments with simulated data

In this section, we evaluate performances of the proposed methods when the data is generated artificially with pre-specified parameters. For this purpose, two different configurations can be proposed, $(n = 10^5, d = 2)$ and $(n = 10^5, d = 5)$ to measure the change in the performances with problem size. To handle data bounds for the sensitivity calculations, both for simulated and real data, we set $\|X\|$ and $\|Y\|$ to the max of the norms over the whole dataset. For each (n, d), data generation is as follows:

- $\theta \sim \mathcal{N}(0, I_d)$, $x_i \sim \mathcal{N}(0, \Sigma_x)$, $\Sigma_x \sim \mathcal{IW}(\Lambda, \kappa)$
- With parameters of $\kappa = d+1$, scale matrix is $\Lambda = V^T V$, and V is a $d \times d$ matrix of i.i.d. variables from $\mathcal{N}(0, 1)$.
- The response variables y are generated with $\sigma_y^2 = 1$.
- For the hyperparameters, we choose same Λ, κ as above and $a = 20$, $b = 0.5$, $m = 0_{d \times 1}$, $C = (a-1)/bI_d$.

According to the comparison results in figure 7.1, algorithms designed for general distributions (`MCMC-fixedS` and `Bayes-fixedS-fast`) outperform `adaSSP` and `MCMC-B&S` in almost all cases both in terms of estimation and prediction. Comparing the full-scale algorithms `MCMC-normalX` and `MCMC-B&S` (that involve updates of S), we observe a clear advantage of `MCMC-normalX` at $d = 2$, but `MCMC-B&S` becomes more competitive at $d = 5$. This can be attributed to the fact that `MCMC-B&S` re-

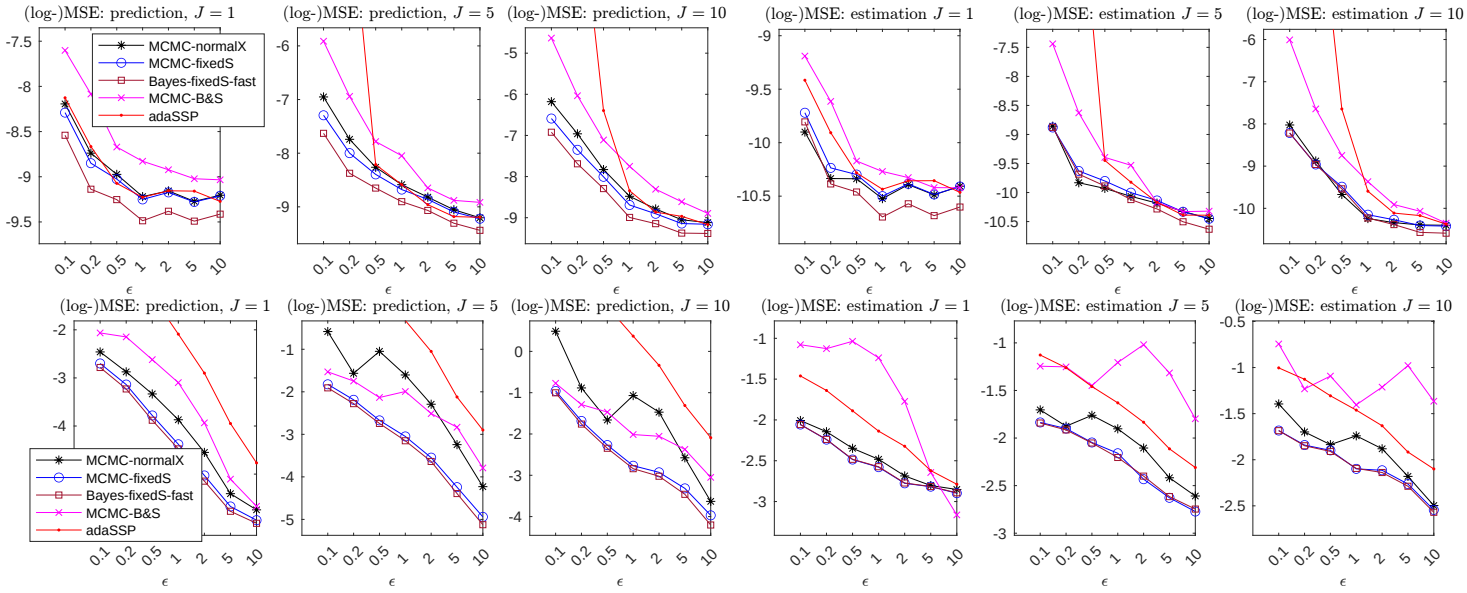

Figure 7.1 Averaged prediction and estimation performances (over 50 runs). Top row: $n = 10^5, d = 2$, Bottom row: $n = 10^5, d = 5$.

quires the extra statistic $y^T y$, unlike `MCMC-normalX`, which causes `MCMC-B&S` to use more noisy statistics. This difference becomes more significant at small d, where the relative effect of the presence of $y^T y$ on the sensitivity is more significant. Finally, all methods improve as ϵ grows, which is expected.

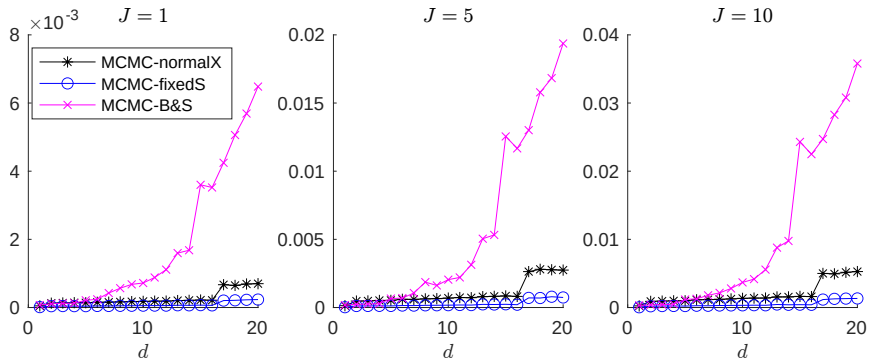

Figure 7.2 Run times per iteration for MCMC algorithms

Computation times per iteration of the MCMC algorithms `MCMC-normalX`, `MCMC-fixedS`, and `MCMC-B&S`[1] according to d can be seen in figure 7.2. It is obvious that computational cost of `MCMC-B&S` is dramatically higher than proposed MCMC algorithms. This is mostly because of the expensive moment calculations in `MCMC-B&S` required for updating the vector of sufficient statistics.

In addition to the MSE, we also consider maximum mean discrepancy (MMD) for evaluating the calibration of the learned posteriors following the method in Bernstein

[1]The algorithms were run in MATLAB 2021b on an Apple M1 chip with 8 cores and 16 GB LPDDR4 memory.

& Sheldon (2019). Roughly, MMD measures the difference between two distributions (Gretton, Borgwardt, Rasch, Schölkopf & Smola, 2012). When, learned posteriors under the privacy constraints similar to the non-private posterior distribution, MMD converges to zero and posteriors are reliable. The (squared) MMD between two distributions can be estimated unbiasedly using i.i.d. samples from those distributions. Non-private and private posteriors for `Bayes-fixedS-fast` are in closed form and can be sampled easily. For the MCMC models, we use every 50th sample of the chain to avoid autocorrelation and thus obtain nearly independent samples. Plots for MMD vs ϵ for each $(J, d = 2)$ is presented in figure 7.3. One can easily see that all the methods converge to the non-private posterior as ϵ increases.

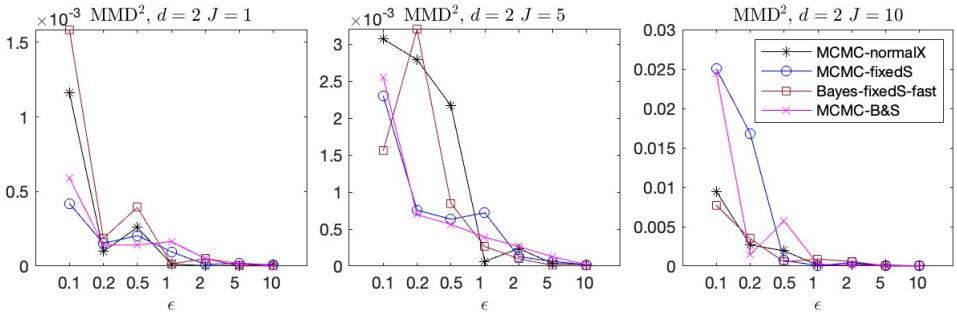

Figure 7.3 Maximum mean discrepancy (MMD) results for each J and $d = 2$.

7.3 Experiments with real data

In addition to the simulated data, performances of the algorithms for the real datasets are also crucially important to evaluate as these methods are mostly going to be utilized for the real problems rather than the experimental settings. In this regard, one of the well-known bases is UCI Machine Learning Repository. Among many datasets, following ones are tested as they mostly include numerical values. Additionally, they have varying data dimensions (d), which makes it possible to observe the performance change for the different data regimes. In the following table, one can check the selected datasets with their acronyms,number of rows and features.

data set	n	d	**hyperlinks**
power plant energy	7655	4	view link
bike sharing	13904	14	view link
air quality	7486	12	view link
3d road	347900	3	view link

During the evaluation, we take 80% of each dataset for the learning and the rest for testing. We present the average prediction performances (out of 50 runs) in Table 7.1 for each dataset and J with $\epsilon = 1$. Note that due to the computational limits, we only present single privacy level with $\epsilon = 1$ since data holder may not want to use smaller ϵ because of the possibility of larger corruption on the data, and similarly higher ϵ values mean less protection, but we observe the similar pattern when we check the results with higher ϵ. We observe that the prediction performances of the compared methods are close, while `MCMC-fixed-S` and `Bayes-fixed-S` are arguably the most stable ones. When $J > 1$ (the distributed data setting), those two methods beat `adaSSP` and `MCMC-B&S` more satisfactorily. Note that we use the same strategy for the data bounds as we utilized in the experiments with simulated data.

Table 7.1 Averaged prediction MSE for the real datasets - $\epsilon = 1$

J	**data sets**	`MCMC-normalX`	`MCMC-fixedS`	`Bayes-fixedS-fast`	`MCMC-B&S`	`adaSSP`
$J = 1$	PowerPlant	0.0129	0.0129	0.0129	**0.0128**	0.0139
	BikeSharing	0.0024	0.0021	0.0021	**0.0020**	0.0107
	AirQuality	0.0060	**0.0057**	**0.0057**	0.0062	0.0066
	3droad	0.0229	0.0229	0.0229	0.0229	0.0229
$J = 5$	PowerPlant	**0.0133**	0.0134	0.0134	0.0136	0.0235
	BikeSharing	0.0174	**0.0045**	**0.0045**	0.0086	0.0382
	AirQuality	0.0142	0.0100	**0.0099**	0.0130	0.0227
	3droad	0.0229	0.0229	0.0229	0.0229	0.0229
$J = 10$	PowerPlant	**0.0142**	0.0143	0.0143	0.0143	0.0351
	BikeSharing	0.0812	**0.0082**	**0.0082**	0.0137	0.0526
	AirQuality	0.0985	**0.0117**	**0.0117**	0.0216	0.0314
	3droad	0.0229	0.0229	0.0229	0.0229	0.0229

Although those values are convincing, it is also important to check whether they are calibrated or not. In other words, are we going to observe similar results when we repeat these experiments for more than 50 runs. For this purpose, one can easily check the confidence intervals for these values. Results for 90% confidence intervals are available in table 7.2. Confidence intervals show that prediction outputs of the algorithms are calibrated and reliable for the evaluation. Hence, proposed methods confidently outperform the state-of-art methods for the private linear regression setting.

Table 7.2 90% CI for prediction MSE for the real datasets - $\epsilon = 1$

J	**data sets**	`MCMC-normalX`	`MCMC-fixedS`	`Bayes-fixedS-fast`	`MCMC-B&S`	`adaSSP`
$J = 1$	PowerPlant	[0.0128,0.0129]	[0.0128,0.0129]	[0.0128,0.0129]	[0.0128,0.0129]	[0.0137,0.0140]
	BikeSharing	[0.0021,0.0027]	[0.0018,0.0024]	[0.0018,0.0024]	[0.0017,0.0022]	[0.0106,0.0108]
	AirQuality	[0.0051,0.0069]	[0.0048,0.0066]	[0.0048,0.0066]	[0.0053,0.0071]	[0.0065,0.0067]
	3droad	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229,0.0229]
$J = 5$	PowerPlant	[0.0132,0.0135]	[0.0132,0.0136]	[0.0132,0.0136]	[0.0135,0.0138]	[0.0234,0.0236]
	BikeSharing	[0.0137,0.0210]	[0.0041,0.0049]	[0.0040,0.0049]	[0.0076,0.0095]	[0.0380,0.0383]
	AirQuality	[0.0109,0.0175]	[0.0089,0.010]	[0.0089,0.0109]	[0.0109,0.0151]	[0.0226, 0.0229]
	3droad	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229, 0.0229]
$J = 10$	PowerPlant	[0.0139,0.0145]	[0.0140,0.0146]	[0.0140,0.0146]	[0.0141,0.0146]	[0.0349,0.0353]
	BikeSharing	[0.0671,0.0954]	[0.0072,0.0092]	[0.0072,0.0092]	[0.0116,0.0158]	[0.0524,0.0527]
	AirQuality	[0.0733,0.1236]	[0.0099,0.0135]	[0.0099,0.0135]	[0.0175,0.0257]	[0.0313,0.0315]
	3droad	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229,0.0229]	[0.0229,0.0229]

8. Conclusion

In this thesis, we mainly discuss applications of differential privacy using Bayesian inference methods. There are two different but complementary perspectives in this work. While one of them is about selection of the best statistic for releasing with noise, The other one focuses on differential private linear regression. Both of them relate Monte Carlo methods with differential privacy.

The statistic selection part analyzes various privacy settings including combinations of the additive statistic or non-additive statistic and the Gaussian mechanism or non-Gaussian mechanism. While additive statistic and Gaussian mechaism enables tractable and closed form calculations, absence of the additivity or Gaussianity requires more sophisticated approaches with Monte Carlo integration. One needs to come up with a Monte Carlo estimation strategy to approximate the value of Fisher information, and advanced MCMC methods for Bayesian inference which enables to work with approximations of the exact posterior distributions for these complicated cases. More importantly, proposed method can work with many types of statistics with only necessary condition of availability of conditional distribution of the generated output given the private statistics. At the end, they are evaluated for various data sharing settings such as normal distribution with unknown variance or uniform distribution with unknown width. One possible limitation of this work occurs when the Fisher information matrix is not indicative for the informativeness of the statistics. For this case, one can use alternative measure such as trace of the Fisher information matrix.

Differing from the statistic selection part, other section of the thesis focuses on coming up with an efficient and effective method for the private linear regression problem. For this purpose, proposed method enables to work with distributed data setting where multiple data holders share their own parts, and injecting less noise to preserve privacy thanks to the novel generative structure. In detail, proposed method aims to sample from the joint posterior distribution using MH-within-Gibbs and sequentially updates model variables throughout the iterations. Methods based on this novel structure outperforms previously developed methods for almost all possible

cases. For the evaluation we use both artifical and real datasets to push methods to their boundaries. However, proposed methods have one limitation mentioned in the remark. Boundedness of normality and requirements of the data bounds for differentially private analysis may be a concern. Fortunately, using the limits from given data as we mentioned is reasonable and utilized by many researchers. Additionally, one may possibly ask which method is the best for which case as there are three effective methods outshining state-of-art methods. `MCMC-normalX` is especially designed for normally distributed features, while `MCMCM-fixedS` and `Bayes-fixedS-fast` are proposed for non-normality. On the other hand, numerical results revelaed that `MCMCM-fixedS` and `Bayes-fixedS-fast` are also competitive under normality. As they are fast and easy-to-implement, we can suggest users try those versions on their first attempts. Between the two, `Bayes-fixedS-fast` is faster but `MCMCM-fixedS` may be safer and provides more insight since it also infers. All that being said, `MCMC-normalX` should not be discarded as it is more capable on exploring P_x.

As for the future extensions, one may improve the estimation strategy for S_j and σ_y^2 while developing methods for the private linear regression methods with non-normality. Also, one future work might be the application of Monte Carlo methods and Bayesian inference on the differentially private gradient optimization problems as they have been intensively utilized by the researchers recently.

BIBLIOGRAPHY

(2010). The random walk metropolis: Linking theory and practice through a case study. *Statistical Science*, *25*(2), 172–190.

Alparslan, B. & Yıldırım, S. (2022). Statistic selection and mcmc for differentially private bayesian estimation. *Statistics and Computing*, *32*(5), 66.

Alparslan, B., Yıldırım, S., & İlker Birbil, (2023). Differentially private distributed bayesian linear regression with mcmc. *Arxiv*, *2301.13778*.

Andrieu, C. & Roberts, G. (2009). The pseudo-marginal approach for efficient monte carlo computations. *Annals of Statistics*, *37*(2), 697 – 725. Publisher: Euclid.

Andrieu, C. & Thoms, J. (2008). A tutorial on adaptive mcmc. *Statistics and Computing*, *18*(4), 343–373.

Andrieu, C. & Vihola, M. (2012). Convergence properties of pseudo-marginal markov chain monte carlo algorithms. *Annals of Applied Probability*, *25*, 1030–1077.

Andrieu, C., Yıldırım, S., Doucet, A., & Chopin, N. (2020). Metropolis-hastings with averaged acceptance ratios. *arXiv*.

Avella-Medina, M. (2021). Privacy-preserving parametric inference: A case for robust statistics. *Journal of the American Statistical Association*, *116*(534), 969–983.

Balle, B. & Wang, Y.-X. (2018). Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*.

Barker, A. A. (1965). Monte carlo calculations of the radial distribution functions for proton-electron plasma. *Australian J. Phys.*, *18*.

Bassily, R., Smith, A. D., & Thakurta, A. (2014). Private empirical risk minimization: Efficient algorithms and tight error bounds. *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, 464–473.

Beaumont, M. A. (2003). Estimation of Population Growth or Decline in Genetically Monitored Populations. *Genetics*, *164*(3), 1139–1160.

Bernstein, G. & Sheldon, D. (2019). *Differentially Private Bayesian Linear Regression*. Red Hook, NY, USA: Curran Associates Inc.

Bernstein, G. & Sheldon, D. R. (2018). Differentially private bayesian inference for exponential families. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Blackwell, D. (1947). Conditional Expectation and Unbiased Sequential Estimation. *The Annals of Mathematical Statistics*, *18*(1), 105 – 110.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konečnỳ, J., Mazzocchi, S., McMahan, B., et al. (2019). Towards federated learning at scale: System design. *Proceedings of machine learning and systems*, *1*, 374–388.

Brooks, S. (1998). Markov chain monte carlo method and its application. *Journal of the Royal Statistical Society: Series D (The Statistician)*, *47*(1), 69–100.

Brooks, S., Gelman, A., Jones, G., & Meng, X.-L. (2011). *Handbook of markov chain monte carlo*. CRC press.

Bun, M. & Steinke, T. (2016). Concentrated differential privacy: Simplifications,

extensions, and lower bounds. In *Proceedings, Part I, of the 14th International Conference on Theory of Cryptography - Volume 9985*, (pp. 635–658)., Berlin, Heidelberg. Springer-Verlag.

Casella, G. & George, E. I. (1992). Explaining the gibbs sampler. *The American Statistician*, *46*(3), 167–174.

Chao, J., Sally Ward, E., & Ober, R. J. (2016). Fisher information theory for parameter estimation in single molecule microscopy: tutorial. *J Opt Soc Am A Opt Image Sci Vis*, *33*(7), B36–57.

Croft, W., Sack, J.-R., & Shi, W. (2022). Differential privacy via a truncated and normalized laplace mechanism. *Journal of Computer Science and Technology*, *37*(2), 369–388.

Dankar, F. K. & Emam, K. E. (2013). Practicing differential privacy in health care: A review. *Trans. Data Priv.*, *6*, 35–67.

Darwish, S. M., Essa, R. M., Osman, M. A., & Ismail, A. A. (2022). Privacy preserving data mining framework for negative association rules: An application to healthcare informatics. *IEEE Access*, *10*, 76268–76280.

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Ranzato, M. a., Senior, A., Tucker, P., Yang, K., Le, Q., & Ng, A. (2012). Large scale distributed deep networks. In Pereira, F., Burges, C., Bottou, L., & Weinberger, K. (Eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Deligiannidis, G., Doucet, A., & Pitt, M. K. (2018). The correlated pseudomarginal method. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *80*(5), 839–870.

Dimitrakakis, C., Nelson, B., Zhang, Z., Mitrokotsa, A., & Rubinstein, B. I. P. (2017). Differential privacy for bayesian inference through posterior sampling. *Journal of Machine Learning Research*, *18*(11), 1–39.

Dong, J., Roth, A., & Su, W. J. (2022). Gaussian differential privacy. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *84*(1), 3–37.

Douc, R., Moulines, É., & Stoffer, D. S. (2013). Nonlinear time series: Theory, methods and applications with r examples. (pp. 494)., New York, NY. Chapman and Hall/CRC.

Drovandi, C. C., Moores, M. T., & Boys, R. J. (2018). Accelerating pseudo-marginal mcmc using gaussian processes. *Computational Statistics Data Analysis*, *118*, 1–17.

Dwork, C. (2008). Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, (pp. 1–19). Springer.

Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In Halevi, S. & Rabin, T. (Eds.), *Theory of Cryptography*, (pp. 265–284)., Berlin, Heidelberg. Springer Berlin Heidelberg.

Dwork, C., Naor, M., Pitassi, T., Rothblum, G. N., & Yekhanin, S. (2010). Pan-private streaming algorithms. In *Proceedings of The First Symposium on Innovations in Computer Science (ICS 2010)*. Tsinghua University Press.

Dwork, C., Roth, A., et al. (2014). The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, *9*(3–4), 211–407.

Dwork, C., Talwar, K., Thakurta, A., & Zhang, L. (2014). Analyze gauss: Optimal bounds for privacy-preserving principal component analysis. In *Proceedings of*

the Forty-Sixth Annual ACM Symposium on Theory of Computing, STOC '14, (pp. 11–20)., New York, NY, USA. Association for Computing Machinery.

Dytso, Alex, B. R. P. H. V. & Shamai, S. (2018). Analytical properties of generalized gaussian distributions. *Journal of Statistical Distributions and Applications*, *5*.

Flötteröd, G. & Bierlaire, M. (2013). Metropolis–hastings sampling of paths. *Transportation Research Part B: Methodological*, *48*, 53–66.

Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. (2013). emcee: the mcmc hammer. *Publications of the Astronomical Society of the Pacific*, *125*(925), 306.

Foulds, J. R., Geumlek, J., Welling, M., & Chaudhuri, K. (2016). On the theory and practice of privacy-preserving bayesian data analysis. *ArXiv*, *abs/1603.07294*.

Gelfand, A. E. (2000). Gibbs sampling. *Journal of the American statistical Association*, *95*(452), 1300–1304.

Geman, S. & Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-6*(6), 721–741.

Gilks, W. & Spiegelhalter, D. (1996). *Markov chain Monte Carlo in practice*. Chapman & Hall/CRC.

Gong, R. (2019). Exact inference with approximate computation for differentially private data via perturbations. *J. Priv. Confidentiality*, *12*.

Gopi, S., Lee, Y. T., & Liu, D. (2022). Private convex optimization via exponential mechanism. In *Conference on Learning Theory*, (pp. 1948–1989). PMLR.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., & Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, *13*(25), 723–773.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, *57*(1), 97–109.

Heikkilä, M., Jälkö, J., Dikmen, O., & Honkela, A. (2019). Differentially private markov chain monte carlo. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Hesterberg, T. (1995). Weighted average importance sampling and defensive mixture distributions. *Technometrics*, *37*(2), 185–194.

Higham, N. (1988). Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra and its Applications*, *103*(C), 103–118.

Human, S. (2022). Advanced data protection control (adpc): An interdisciplinary overview. *arXiv preprint arXiv:2209.09724*.

Isaak, J. & Hanna, M. J. (2018). User data privacy: Facebook, cambridge analytica, and privacy protection. *Computer*, *51*(8), 56–59.

Ju, N., Awan, J., Gong, R., & Rao, V. (2022a). Data augmentation MCMC for bayesian inference from privatized data. In Oh, A. H., Agarwal, A., Belgrave, D., & Cho, K. (Eds.), *Advances in Neural Information Processing Systems*.

Ju, N., Awan, J. A., Gong, R., & Rao, V. A. (2022b). Data augmentation mcmc for bayesian inference from privatized data. *arXiv preprint arXiv:2206.00710*.

Kasiviswanathan, S. P., Lee, H. K., Nissim, K., Raskhodnikova, S., & Smith, A. (2008). What can we learn privately? In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, (pp. 531–540).

Kharkovskii, D., Dai, Z., & Low, B. K. H. (2020). Private outsourced Bayesian optimization. In III, H. D. & Singh, A. (Eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, (pp. 5231–5242). PMLR.

Kifer, D., Smith, A., & Thakurta, A. (2012). Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*, (pp. 25–1). JMLR Workshop and Conference Proceedings.

Kleijn, B. J. & Van der Vaart, A. W. (2012). The bernstein-von-mises theorem under misspecification.

Kuru, N., Birbil, S. I., Gürbüzbalaban, M., & Yıldırım, S. (2020). Differentially private accelerated optimization algorithms. *SIAM J. Optim.*, *32*, 795–821.

Kusner, M. J., Gardner, J. R., Garnett, R., & Weinberger, K. Q. (2015). Differentially private bayesian optimization. In *International Conference on Machine Learning*.

Lehmann, E. & D'Abrera, H. (1998). *Nonparametrics: Statistical Methods Based on Ranks*. Prentice Hall.

Li, B., Chen, C., Liu, H., & Carin, L. (2019). On connecting stochastic gradient mcmc and differential privacy. In *The 22nd International Conference on Artificial Intelligence and Statistics*, (pp. 557–566). PMLR.

Liu, F. (2019). Generalized gaussian mechanism for differential privacy. *IEEE Transactions on Knowledge and Data Engineering*, *31*(4), 747–756.

Liu, J. S., Wong, W. H., & Kong, A. (1994). Covariance structure of the gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika*, *81*(1), 27–40.

Liu, X., Jain, P., Kong, W., Oh, S., & Suggala, A. (2023). Near optimal private and robust linear regression.

Malagò, L. & Pistone, G. (2015). Information geometry of the gaussian distribution in view of stochastic optimization. In *Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII*, FOGA '15, (pp. 150–162)., New York, NY, USA. Association for Computing Machinery.

Martino, L., Read, J., & Luengo, D. (2015). Independent doubly adaptive rejection metropolis sampling within gibbs sampling. *IEEE Transactions on Signal Processing*, *63*(12), 3123–3138.

Matte, C., Bielova, N., & Santos, C. (2020). Do cookie banners respect my choice? measuring legal compliance of banners from iab europe's transparency and consent framework.

McSherry, F. & Talwar, K. (2007). Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, (pp. 94–103).

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, *21*(6), 1087–1092.

Millar, R. B. & Meyer, R. (2000). Non-linear state space modelling of fisheries biomass dynamics by using metropolis-hastings within-gibbs sampling. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *49*(3), 327–342.

Myers, R. H. & Montgomery, D. C. (1997). A tutorial on generalized linear models. *Journal of Quality Technology*, *29*(3), 274–291.

Nissim, K., Raskhodnikova, S., & Smith, A. (2007). Smooth sensitivity and sampling in private data analysis. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, (pp. 75–84)., New York, NY, USA. Association for Computing Machinery.

Oberski, D. L. & Kreuter, F. (2020). Differential privacy and social science: An urgent puzzle. *Harvard Data Science Review*, *2*(1), 1.

Park, T. & Lee, S. (2022a). Improving the gibbs sampler. *Wiley Interdisciplinary Reviews: Computational Statistics*, *14*(2), e1546.

Park, T. & Lee, S. (2022b). Improving the gibbs sampler. *WIREs Computational Statistics*, *14*(2), e1546.

Peskun, P. H. (1973). Optimum monte-carlo sampling using markov chains. *Biometrika*, *60*(3), 607–612.

Pfeiffer, R. M., Kapla, D. B., & Bura, E. (2021). Least squares and maximum likelihood estimation of sufficient reductions in regressions with matrix-valued predictors. *International Journal of Data Science and Analytics*, *11*(1), 11–26.

Räisä, O., Koskela, A., & Honkela, A. (2021). Differentially private hamiltonian monte carlo. *ArXiv*, *abs/2106.09376*.

Rajaratnam, B. & Sparks, D. (2015). Mcmc-based inference in the era of big data: A fundamental analysis of the convergence complexity of high-dimensional chains. *arXiv: Statistics Theory*.

Robert, C. P. & Casella, G. (2004). *Diagnosing Convergence*, (pp. 459–509). New York, NY: Springer New York.

Robert, C. P. & Roberts, G. O. (2021). Rao-blackwellization in the mcmc era.

Ryffel, T., Bach, F., & Pointcheval, D. (2022). Differential privacy guarantees for stochastic gradient langevin dynamics. *arXiv preprint arXiv:2201.11980*.

Schervish, M. J. (1995). *Sufficient Statistics*, (pp. 111). New York, NY: Springer New York.

Sharma, S. (2017). Markov chain monte carlo methods for bayesian data analysis in astronomy. *arXiv preprint arXiv:1706.01629*.

Smith, A. F. M. (1991). Bayesian computational methods. *Philosophical Transactions of the Royal Society of London. Series A: Physical and Engineering Sciences*, *337*, 369 – 386.

Spade, D. A. (2020). Chapter 1 - markov chain monte carlo methods: Theory and practice. In A. S. Srinivasa Rao & C. Rao (Eds.), *Principles and Methods for Data Science*, volume 43 of *Handbook of Statistics* (pp. 1–66). Elsevier.

Sun, L., Zhou, Y., Yu, P. S., & Xiong, C. (2020). Differentially private deep learning with smooth sensitivity. *ArXiv*, *abs/2003.00505*.

Tokdar, S. T. & Kass, R. E. (2010). Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, *2*(1), 54–60.

Trautman, L. J. (2022). Tik tok! tiktok: Escalating tension between u.s. privacy rights and national security vulnerabilities.

Triantafyllopoulos, K. (2002). Moments and cumulants of the multivariate real and complex gaussian distributions. Department of Mathematics, University of Bristol.

Vaart, A. W. v. d. (1998). *Asymptotic Statistics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press.

Vadhan, S. & Wang, T. (2021). Concurrent composition of differential privacy. In Nissim, K. & Waters, B. (Eds.), *Theory of Cryptography*, (pp. 582–604).,

Cham. Springer International Publishing.

Vapnik, V. N. (1991). Principles of risk minimization for learning theory. In *NIPS*.

Varshney, P., Thakurta, A., & Jain, P. (2022). (nearly) optimal private linear regression for sub-gaussian data via adaptive clipping. In Loh, P.-L. & Raginsky, M. (Eds.), *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, (pp. 1126–1166). PMLR.

Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., & Rellermeyer, J. S. (2020). A survey on distributed machine learning. *Acm computing surveys (csur)*, *53*(2), 1–33.

Walsh, B. (2004). Markov chain monte carlo and gibbs sampling. *Lecture Notes for EEB 581, version 26, April*.

Wang, D., Chen, C., & Xu, J. (2019). Differentially private empirical risk minimization with non-convex loss functions. In *International Conference on Machine Learning*, (pp. 6526–6535). PMLR.

Wang, Y.-X. (2018). Revisiting differentially private linear regression: optimal and adaptive prediction & estimation in unbounded domain. *ArXiv*, *abs/1803.02596*.

Wang, Y.-X., Fienberg, S., & Smola, A. (2015). Privacy for free: Posterior sampling and stochastic gradient monte carlo. In Bach, F. & Blei, D. (Eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, (pp. 2493–2502)., Lille, France. PMLR.

Warne, D. J., Baker, R. E., & Simpson, M. J. (2020). A practical guide to pseudo-marginal methods for computational inference in systems biology. *Journal of Theoretical Biology*, *496*, 110255.

Wilson, A. G. & Ghahramani, Z. (2011). Generalised wishart processes. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, (pp. 736–744)., Arlington, Virginia, USA. AUAI Press.

Yıldırım, S. & Ermiş, B. (2019). Exact mcmc with differentially private moves. *Statistics and Computing*, *29*(5), 947–963.

Yıldırım, S., Andrieu, C., & Doucet, A. (2018). Scalable monte carlo inference for state-space models. *arXiv*.

Zhang, J., Zhang, Z., Xiao, X., Yang, Y., & Winslett, M. (2012). Functional mechanism: Regression analysis under differential privacy. *Proc. VLDB Endow.*, *5*(11), 1364–1375.

Zhang, Q., Bu, Z., Chen, K., & Long, Q. (2021). Differentially private bayesian neural networks on accuracy, privacy and reliability. *ArXiv*, *abs/2107.08461*.

Zhang, Y. & Lin, X. (2015). Disco: Distributed optimization for self-concordant empirical loss. In Bach, F. & Blei, D. (Eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, (pp. 362–370)., Lille, France. PMLR.

Zhang, Z., Rubinstein, B. I. P., & Dimitrakakis, C. (2016). On the differential privacy of bayesian inference. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, (pp. 2365–2371). AAAI Press.

Zhao, Y. & Chen, J. (2022a). A survey on differential privacy for unstructured data content. *ACM Computing Surveys (CSUR)*, *54*(10s), 1–28.

Zhao, Y. & Chen, J. (2022b). A survey on differential privacy for unstructured data content. *ACM Comput. Surv.*, *54*(10s).

Zhu, T. & Yu, P. S. (2019). Applying differential privacy mechanism in artificial intelligence. *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 1601–1609.