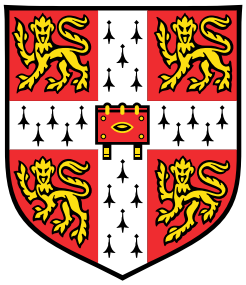
Differential Privacy & Approximate Bayesian Inference



Mrinank Sharma

Supervisor: Dr Richard E. Turner

Department of Engineering
University of Cambridge

This report is submitted for the degree of
Master of Engineering

Pembroke College

May 2019

Abstract

ms2314: Need to write this

Table of contents

Chapter 1

Introduction

Machine learning methods are trained on large quantities of data, leveraging information within the dataset in order to make predictions about previously unseen data and make decisions. Recently, such methods have found use in scenarios where the data used in personal and sensitive, one example being the use of human genomic data to predict drug sensitivity Niinimäki et al. [2019]. Typically, data relating to individuals in training datasets are *anonymised*, for example, by removing all identifiable information (such as names, addresses, etc) and replacing this information with an anonymous identifier. However, Narayanan and Shmatikov show that anonymisation is insufficient, partially due to the availability of *auxiliary information* i.e. additional, publicly available information. When Netflix released a dataset in 2006 containing movie ratings for approximately 500000 subscribers with names replaced with identifiers, the data could be combined with public ratings on Internet Movie Database to identify movie ratings of two users. Narayanan and Shmatikov [2008] Intuitively, data-points about individuals are highly dimensional meaning that anonymisation is insufficient, a further example being that even when sharing DNA sequence data without identifiers, it is possible to recover particular surnames using additional metadata. Gymrek et al. [2013]

ms2314: Look into these papers more

Whilst a particular user's film ratings are not particularly sensitive, a lack of privacy in the areas of healthcare and public policy are critical.

Fredrikson et. al have shown that *model inversion attacks* are possible, where an adversary seeks to learn information about training data given model predictions. In particular, a neural network for facial recognition which returned confidence values was exploited in order to recover the image of a training set participant. Whilst more sophisticated attacks will be required for algorithms which do not provide confidence information, it is therefore possible for an adversary to recover anonymised training data-points and then de-anonymise this information using auxiliary information. Fredrikson et al. [2015]

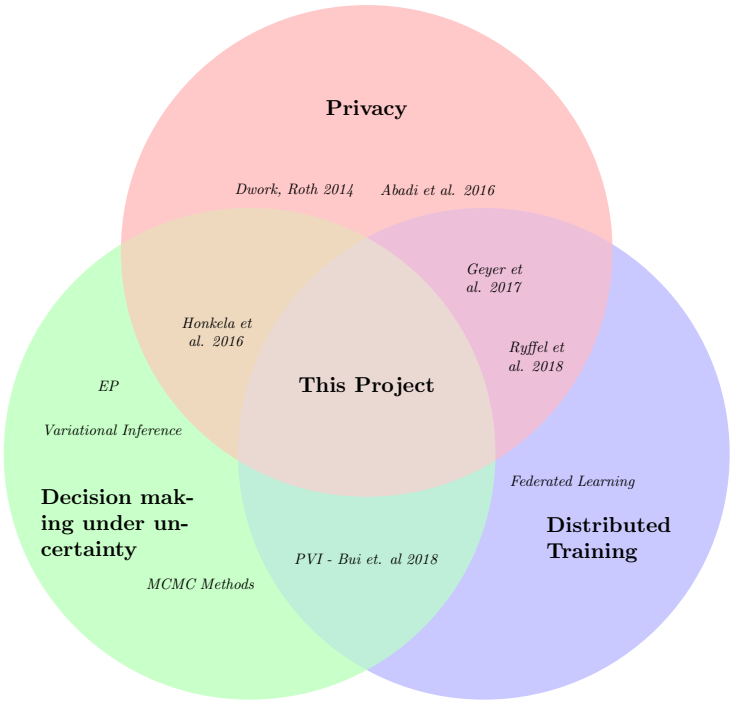


Fig. 1.1 Project aims, including other work within this area.

The increasing availability and affordability of mobile smart-phones means that the *federated learning* context, where data across a number of clients is used to train a global model without transferring local data to a central server, is particularly interesting. Additionally, federated learning schemes reduce power consumption and intuitively give stronger privacy by removing the requirement of transferring entire local datasets. goo [2017] Additionally, in order to make optimal decisions, it is important to model the uncertainty in what is known. This corresponds to performing Bayesian inference and producing a *posterior distribution* over unknown variables.

ms2314: Update Venn Diagram - also add these papers in the references of this report

This project aims to develop a generalised method to enable Bayesian inference to be performed in contexts where data is distributed over a number of clients whilst also providing privacy guarantees for each client.

Chapter 2

Literature Review

2.1 Differential Privacy

2.1.1 Preliminaries

Differential privacy is a mathematical technique which formalises privacy and is able to numerically quantify the level of privacy that some method provides. This is particularly useful not only from the point of view of a designer who is able to compare techniques formally but also from the point of view of a client whose data we seek to protect as this formalism enables them to choose particular settings corresponding to the level of privacy that they seek.

ms2314: Perhaps add diagram showing privacy barrier / we generally assume that the adversary is able to do anything to the data after some sort of model has been released.

Definition 2.1.1 (ϵ-Differential Privacy) *A randomized algorithm, $\mathcal{A}$, is said to be ϵ-differentially private if for any possible subset of outputs, S, and for all pairs of datasets, $(\mathcal{D}, \mathcal{D}')$, which differ in one entry only, the following inequality holds:*

$$Pr(\mathcal{A}(\mathcal{D}) \in S) \leq e^{\epsilon} Pr(\mathcal{A}(\mathcal{D}') \in S) \tag{2.1}$$

Thus, differential privacy provides privacy in the sense that the output probability densities ought to be similar (i.e. bounded by e^{ϵ}) for datasets which are also similar (i.e. differ in only entry only). ϵ, a positive quantity, quantifies the level of privacy provided; large values of epsilon allow the resulting output densities to differ significantly whilst small values of epsilon mean that the output densities are similar. Dwork and Roth [2014]

The key intuition behind differential privacy, noting the requirement that the algorithm is **randomized**, is to introduce noise which obscures the contribution of any particular data-point meaning that any adversary is unable to determine whether a particularly output was simply due to noise or due to a specific data-point.

Often, the above definition of differential privacy is slackened by introducing an extra privacy variable.

Definition 2.1.2 ((ϵ, δ)-Differential Privacy) *A randomized algorithm, $\mathcal{A}$, is said to be (ϵ, δ) differentially private if for any possible subset of outputs, S, and for all datasets, $(\mathcal{D}, \mathcal{D}')$, which differ in one entry only, the following inequality holds:*

$$Pr(\mathcal{A}(\mathcal{D}) \in S) \leq e^{\epsilon} Pr(\mathcal{A}(\mathcal{D}') \in S) + \delta \tag{2.2}$$

Similar to ϵ, larger values of δ correspond to weaker privacy guarantees and $\delta = 0$ corresponds to a pure ϵ-differentially private algorithm. Note that it can be shown that (ϵ, δ)-DP provides a probabilistic ϵ-DP guarantee with probably $1 - \delta$. Additionally, it can be shown that differential privacy is immune to *post-processing* i.e. without additional knowledge, it is not possible to reduce the level of privacy provided by a differentially private algorithm. Dwork and Roth [2014]

A useful quantity relating to a function of some dataset is the ℓ_2 sensitivity.

Definition 2.1.3 (ℓ_2 Sensitivity) *The ℓ_2 sensitivity of function $f : \mathcal{D} \rightarrow \mathbb{R}^n$, is denoted as $\Delta_2(f)$ and is defined as:*

$$\Delta_2(f) = \max_{\mathcal{D}, \mathcal{D}'} ||f(\mathcal{D}) - f(\mathcal{D}')||_2 \tag{2.3}$$

where $\mathcal{D}$ and $\mathcal{D}'$ differ in one entry only.

For a function with a given ℓ_2 sensitivity, the Gaussian mechanism can be used to provide an (ϵ, δ)-differential privacy guarantee.

Theorem 2.1.4 (Gaussian Mechanism) *Let $f : \mathcal{D} \rightarrow \mathbb{R}^n$ be a function with ℓ_2 sensitivity $\Delta_2(f)$. Releasing $f(\mathcal{D}) + \eta$ is (ϵ, δ)-differentially private where $\eta \sim \mathcal{N}(0, \sigma^2)$ when:*

$$\sigma^2 > 2\ln(1.25/\delta)\Delta_2^2(f)/\epsilon^2 \tag{2.4}$$

when $\epsilon \in (0, 1)$.

Dwork and Roth [2014]

Typically, a particular process may be repeatedly applied to a dataset; a simple example being that in many machine learning problems, a loss function can be composed into a sum over data-points and gradient descent techniques repeatedly compute the gradient of the loss over all of the data-points, repeatedly updating the gradient to reduce the loss. Ruder [2016] Therefore, it is essential to be able to *compose* the total loss of privacy (in terms of an overall value for ϵ and δ) when a randomised algorithm is repeatedly applied.

Fortunately, there exist many schemes which allow values of ϵ and δ to be computed. The *Moments Accountant* is an advanced technique which is able to 'account' for and track the total privacy expenditure of a technique by tracking the moments of the privacy loss.

2.1.2 The Moments Accountant

Definition 2.1.5 (Privacy Loss) *For neighbouring datasets, $\mathcal{D}$ and $\mathcal{D}'$, a stochastic algorithm, $\mathcal{A}$ and some outcome, S, define the privacy loss at S as*

$$c(S;\mathcal{A},\mathcal{D},\mathcal{D}') \triangleq \log \frac{\Pr[\mathcal{A}(\mathcal{D})=S]}{\Pr[\mathcal{A}(\mathcal{D}')=S]} \tag{2.5}$$

Intuitively, the privacy loss is large when the probability of the observed outcome is very different across neighbouring datasets, corresponding to a large absolute value of c. Note that since the algorithms we use are stochastic, the privacy loss is a random variable. Abadi et al. [2016]

ms2314: Define differential privacy as a bound on privacy loss

Definition 2.1.6 (λth Moment of $\mathcal{A}$) *For algorithm $\mathcal{A}$, a quantity of interest is the maximum value of the log of the moment generating function of the privacy loss.*

$$\alpha_{\mathcal{A}}(\lambda) \triangleq \max_{\mathcal{D},\mathcal{D}'} \log \mathbb{E}_{S\sim\mathcal{A}(\mathcal{D})}\Big\{\exp \lambda c(S;\mathcal{A},\mathcal{D},\mathcal{D}')\Big\} \tag{2.6}$$

Abadi et al. [2016]

Theorem 2.1.7 (Properties of $\alpha_{\mathcal{A}}(\lambda)$) *$\alpha_{\mathcal{A}}(\lambda)$ exhibits two important properties.*

1. ***Composability:*** *If the algorithm $\mathcal{A}$ consists of a sequence of adaptive mechanisms $\mathcal{M}_1,\ldots,\mathcal{M}_k$, for any λ:*
$$\alpha_{\mathcal{A}}(\lambda) \leq \sum_{i=1}^{k} \alpha_{\mathcal{M}_i}(\lambda) \tag{2.7}$$
2. ***Tail Bound:*** *For any $\epsilon > 0$ and λ, the stochastic algorithm, $\mathcal{A}$, is (ϵ,δ)-differentially private for*
$$\delta \leq \exp(\alpha_{\mathcal{A}}(\lambda) - \lambda\epsilon) \tag{2.8}$$

The moments accountant scheme computes $\alpha_\alpha(\mathcal{M}_i)$ for each step for several values of λ and uses the composability property to compute the aggregate loss. Abadi et al. [2016] In typical applications, either the target value of ϵ or δ is fixed at some value, and thus the tail bound property is applied to compute the best possible value of the other privacy variable, using the appropriate following equation:

$$\delta = \min_{\lambda} \exp(\alpha_{\mathcal{A}}(\lambda) - \lambda\epsilon) \tag{2.9}$$

$$\epsilon = \min_{\lambda} \frac{1}{\lambda}(\alpha_{\mathcal{A}}(\lambda) - \ln\delta) \tag{2.10}$$

Let $\mathcal{D} = \{\boldsymbol{x}_i\}$ and $\mathcal{D}' = \mathcal{D} \cup \boldsymbol{x}'$ where $\boldsymbol{x} \in \mathcal{X}$. Let $f : \mathcal{X} \to \mathbb{R}^n$ with $||f(\cdot)||_2 \leq \Delta$. Consider the following mechanism, known as the *Gaussian Mechanism with sub-sampling*

$$\mathcal{M}(\mathcal{D}) = \sum_{i \in J} f(\boldsymbol{x}_i) + \mathcal{N}(\boldsymbol{0}, \Delta^2 \sigma^2 \boldsymbol{I}) \tag{2.11}$$

where i is a subset of indices where each index is chosen independently with probability q. Without loss of generality, let $f(\boldsymbol{x}_i) = \boldsymbol{0}$ and $f(\boldsymbol{x}') = \Delta \cdot \boldsymbol{e}$ where $\boldsymbol{e}_1$ is a unit vector. Then $\mathcal{M}(\mathcal{D})$ and $\mathcal{M}(\mathcal{D}')$ are distributed identically other than the coordinate corresponding to $\boldsymbol{e}_1$. Considering only this direction, the output densities for the mechanism are:

$$\mathcal{M}(\mathcal{D}) \sim \mu_0 \triangleq \mathcal{N}(0, \Delta^2 \sigma^2) \tag{2.12}$$

$$\mathcal{M}(\mathcal{D})' \sim \mu_1 \triangleq q \cdot \mathcal{N}(\Delta, \Delta^2 \sigma^2) + (1 - q) \cdot \mathcal{N}(0, \Delta^2 \sigma^2) \tag{2.13}$$

Then, $\alpha_{\mathcal{M}}(\lambda)$ can be computed as:

$$\alpha_{\mathcal{M}}(\lambda) = \ln \max(E_1, E_2) \tag{2.14}$$

$$E_1 = \mathbb{E}_{z \sim \mu_0}\left[\left(\frac{\mu_0(z)}{\mu_1(z)}\right)^{\lambda}\right] \tag{2.15}$$

$$E_2 = \mathbb{E}_{z \sim \mu_1}\left[\left(\frac{\mu_1(z)}{\mu_0(z)}\right)^{\lambda}\right] \tag{2.16}$$

where each integral can be evaluated using numerical integration. Both integrals must be considered since the maximum of the log moments of the privacy loss must be found, and both $\mathcal{D}$ and $\mathcal{D}'$ could be considered as the 'original' dataset. Abadi et al. [2016]

Note: the computed values of ϵ and δ are independent of the data used for the mechanism and could be pre-computed given fixed values of Δ, q and either δ or ϵ.

2.1.3 Differentially Private Stochastic Gradient Descent

Abadi et. al propose differentially private stochastic gradient descent by adapting stochastic gradient descent to use the Gaussian mechanism with sub-sampling to compute the gradient of some loss function. Abadi et al. [2016]

Algorithm 1 outlines the differentially private stochastic gradient descent method. Note that the gradient clipping used in this method effectively limits the contribution of each data-points towards the gradient, bounding the ℓ_2 sensitivity and enabling the Gaussian mechanism with sub-sampling to be applied. Noting that the noise added scales with the clipping bound, large values of the clipping bound correspond to strong, but noisy gradient signals. Abadi et. al suggest setting c_t to be fixed at the median of the norms of unclipped gradients throughout the course of training as well as using a learning rate with starts off relatively large and is reduced over time. Additionally, it is remarked that $L \simeq \sqrt{N}$ is

appropriate; larger values of L improve the accuracy of the gradient signal but increase the privacy cost. Abadi et al. [2016]

ms2314: Haven't mentioned privacy amplification - perhaps I outhg to here

Algorithm 1 Differentially Private Stochastic Gradient Descent (DP-SGD)

Input: Dataset $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, Loss function $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N}\sum_i \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}_i)$
Parameters: Learning Rate η_t, Clipping Bound c_t, Lot Size L, DP Noise Scale σ_t, Num. Iterations T

Initialise $\boldsymbol{\theta}_0$ randomly.
for $t = 1, \ldots, T$ **do**
  Take a random sample L_t with sampling probability $q = L/N$
  for all $i \in L_t$ **do** ▷ Compute Gradient
    $\boldsymbol{g}_t(\boldsymbol{x}_i) \leftarrow \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}_i)$
    $\tilde{\boldsymbol{g}}_t(\boldsymbol{x}_i) \leftarrow \boldsymbol{g}_t(\boldsymbol{x}_i) / \max(||\boldsymbol{g}_t(\boldsymbol{x}_i)||_2 / c_t, 1)$ ▷ Gradient Clipping
  end for
  $\tilde{\boldsymbol{g}}_t \leftarrow \frac{1}{L}[\sum_{i \in L_t} \boldsymbol{g}_t(\boldsymbol{x}_i) + \mathcal{N}(0, \sigma_t^2 c_t^2 \boldsymbol{I})]$ ▷ Perturb Clipped Gradient
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \tilde{\boldsymbol{g}}_t$
end for
Output $\boldsymbol{\theta}_T$ and calculate (ϵ, δ) using the Moments Accountant.

In Abadi et al. (2016), this technique was applied to deep neural networks. However, this algorithm can be used as a building block to create other differentially private techniques, and indeed this technique has been applied in order to perform variational inference for non conjugate models Jälkö et al. [2016].

2.2 Federated Learning

2.2.1 Partitioned Variational Inference

Partitioned Variational Inference (PVI) is a general framework which encompasses many variational Bayesian techniques. Assume that the dataset, $\mathcal{D}$ has been partitioned into M shards i.e. $\mathcal{D} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_M\}$, where $\boldsymbol{X}_i = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_i}\}$. A probabilistic model with parameters $\boldsymbol{\theta}$ has been suggested to model this data with a known prior, $p(\boldsymbol{\theta})$ and likelihood function, $p(\boldsymbol{x}|\boldsymbol{\theta})$. The aim of Bayesian inference is to calculate the posterior density over the parameters, $p(\boldsymbol{\theta}|\mathcal{D})$ but in general, it is not possible to compute this distribution. In variational methods, an approximate distribution, $q(\theta)$, is used to approximate the posterior. In PVI, the proposal distribution takes the form:

$$q(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{m=1}^{M} t_m(\boldsymbol{\theta}) \simeq \frac{1}{\mathcal{Z}} p(\boldsymbol{\theta}) \prod_{m=1}^{M} p(\boldsymbol{X}_m|\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\mathcal{D}) \tag{2.17}$$

Inspecting the above equation, it can be seen that each $t_m(\boldsymbol{\theta})$ approximates the (unnormalised) likelihood $p(\boldsymbol{X}_m|\boldsymbol{\theta})$, noting that the approximate posterior does not include a normalising constant.

Algorithm 2 Partitioned Variational Inference (PVI)

Input: Partitioned Dataset $\mathcal{D} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_M\}$, Prior $p(\boldsymbol{\theta})$, Family of Distributions $\mathcal{Q}$

Initialise approximate likelihood:

$$t_m^{(0)}(\boldsymbol{\theta}) \leftarrow 1 \ \forall m \tag{2.18}$$

Initialise approximate posterior:

$$q^{(0)}(\boldsymbol{\theta}) \leftarrow p(\boldsymbol{\theta}) \tag{2.19}$$

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

  $b_i \leftarrow$ index of next approximate likelihood to update.

  Compute the new approximate likelihood:

$$q^{(i)}(\boldsymbol{\theta}) \leftarrow \underset{q(\boldsymbol{\theta})\in\mathcal{Q}}{\operatorname{argmax}} \int q(\boldsymbol{\theta}) \ln \frac{q^{(i-1)}(\boldsymbol{\theta}) p(\boldsymbol{X}_{b_i}|\boldsymbol{\theta})}{q(\boldsymbol{\theta}) t_{b_i}^{(i-1)}(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \tag{2.20}$$

  Update the approximate likelihood for the updated factor:

$$t_{b_i}^{(i)}(\boldsymbol{\theta}) \leftarrow \frac{q^{(i)}(\boldsymbol{\theta})}{q^{(i-1)}(\boldsymbol{\theta})} t_{b_i}^{(i-1)}(\boldsymbol{\theta}) \tag{2.21}$$

end for

When partitioned variational inference is used in the federated learning context, each client stores its own data partition, $\boldsymbol{X}_m$, and communicated with a central parameter server which stores the global approximate posterior. At each stage, each client maximises a *local* free energy. Bui et al. [2018]

Definition 2.2.1 (Local Free Energy) *The local free energy is defined as:*

$$\mathcal{F}_{b_i}^{(i)}(q(\boldsymbol{\theta})) = \int q(\boldsymbol{\theta}) \ln \frac{q^{(i-1)}(\boldsymbol{\theta}) p(\boldsymbol{X}_{b_i}|\boldsymbol{\theta})}{q(\boldsymbol{\theta}) t_{b_i}^{(i-1)}(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \tag{2.22}$$

It can be shown that the local free energy optimisation (Eq. 2.20) is equivalent to the following $\mathcal{KL}$ optimisation:

$$q^{(i)}(\boldsymbol{\theta}) \leftarrow \underset{q(\boldsymbol{\theta})\in\mathcal{Q}}{\operatorname{argmax}} \mathcal{KL}(q(\boldsymbol{\theta}) || \hat{p}_{b_i}^{(i)}(\boldsymbol{\theta})) \tag{2.23}$$

where $\hat{p}_{b_i}^{(i)}(\boldsymbol{\theta})$ is known as the *tilted distribution* and is defined as:

$$\hat{p}_{b_i}^{(i)}(\boldsymbol{\theta}) = \frac{1}{\mathcal{Z}_{b_i}^{(i)}} \frac{q^{(i-1)}(\boldsymbol{\theta})}{t_{b_i}^{(i-1)}(\boldsymbol{\theta})} p(\boldsymbol{X}_{b_i}|\boldsymbol{\theta}) \tag{2.24}$$

$$= \frac{1}{\mathcal{Z}_{b_i}^{(i)}} p(\boldsymbol{\theta}) p(\boldsymbol{X}_{b_i}|\boldsymbol{\theta}) \prod_{m \neq b_i} t_m^{(i-1)}(\boldsymbol{\theta}) \tag{2.25}$$

which can be interpreted as a local estimate of the posterior for client b_i, using the exact local likelihood and approximate likelihood terms for the data partitions held at other clients.

In standard variational inference, a global free energy, depending on the entire dataset is maximised.

Definition 2.2.2 (Global Free Energy) *The global free energy is defined as:*

$$\mathcal{F}(q(\boldsymbol{\theta})) = \int q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{\theta}) \prod_{m=1}^{M} p(\boldsymbol{X}_m|\boldsymbol{\theta})}{q(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \tag{2.26}$$

Maximising this free energy is equivalent to minimising the $\mathcal{KL}$ divergence between the approximate posterior and the true posterior. Bishop [2006]

Now we return to the partitioned variational inference setting.

Theorem 2.2.3 (Properties of PVI) *Consider the approximate posterior at convergence* $q^*(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{m=1}^{M} t_m^*(\boldsymbol{\theta})$. *Then:*

1. *The sum of local free energies is the global free energy:*

$$\sum_{m=1}^{M} \mathcal{F}_m(q^*(\boldsymbol{\theta})) = \mathcal{F}(q^*(\boldsymbol{\theta})) \tag{2.27}$$

2. *Optimising the local free energies optimises the global free energy:*

$$q^*(\theta) = \underset{q(\boldsymbol{\theta}) \in \mathcal{Q}}{\operatorname{argmax}} \, \mathcal{F}_m(q(\boldsymbol{\theta})) \; \forall m \Rightarrow q^*(\theta) = \underset{q(\boldsymbol{\theta}) \in \mathcal{Q}}{\operatorname{argmax}} \, \mathcal{F}(q(\boldsymbol{\theta})) \tag{2.28}$$

The above properties of PVI motivate the scheme and suggest that it may provide reasonable result. Bui et al. [2018]

ms2314: perhaps remove this, may have gone into too much depth here.

Chapter 3

Differentially Private Partitioned Variational Inference

In this chapter, we construct different forms of *Differentially Private Partitioned Variational Inference (DP-PVI)*, considering the full problem context and implications with regards to strength of privacy protection. This is a key contribution of this work.

3.1 Context

The *adversary* aims to use accessible information to glean private and sensitive information which the machine learning model has been trained with. Recalling the PVI setting, there are a number of clients with access to their own local data shards containing sensitive information. In the fully general context, it is assumed that:

- The adversary has full, unlimited access to the approximate posterior, $q(\boldsymbol{\theta})$, including not only access to full trained approximate posterior but also access to approximate posterior throughout the course of training.
- The adversary is able to intercept messages between the central parameter server and each client.
- The adversary is able to plant concealed 'enemy' clients or coerce existing clients to reveal their local data and/or plant data which is to be used to train the global approximate model.

Fig. 3.1 summarises the fully generalised context of this project.

Different methods of adapting PVI will have different implications as to which parties are or are not trusted and may have asymmetric privacy guarantees, conferring different levels of privacy depending on what the specific adversary is able to access. We will remark on these implications in the following sections.

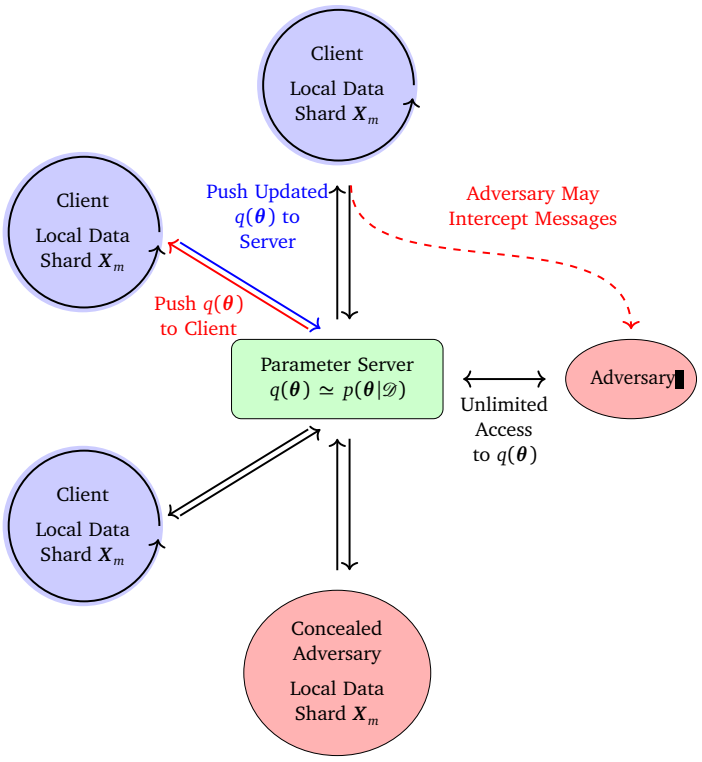


Fig. 3.1 Generalised Project Context Summary: there are a number of clients with local, sensitive data shards, $\{\boldsymbol{X}_m\}$. It is assumed that the adversary has unlimited access to the global model, $q(\boldsymbol{\theta})$, and the adversary may be able to intercept messages.

3.2 Forms of DP-PVI

ms2314: Perhaps reword for clarity - not entirely convinced with this

3.2.1 Datapoint Level DP-PVI

A simple method to construct DP-PVI is to perform the local free energy optimisation (Eq. 2.20) in a privacy preserving manner, for example using DP-SGD (Algorithm 1) with the loss function given by the negative local free energy. If this scheme were to be used, the client is projected against parties as any external communication, namely approximate posterior updates, will have been produced using a method which protects the data-points in each shard.

For clarity, assume that a data-shard is comprised of a number of data-points i.e. $\boldsymbol{X}_m = \{\boldsymbol{x}_1^{(m)}, \ldots, \boldsymbol{x}_{N_m}^{(m)}\}$. This scheme would limit the contribution of each $\boldsymbol{x}_i^{(m)}$ to the update of the approximate posterior and add noise corresponding to the the maximum possible contribution of each data-point.

Advantage of this schemes include there being no requirement for encryption on outgoing client messages and no authentication requirements for the client; the client would not need to ensure that it is indeed communicating with the a trustworthy parameter server i.e. even if the parameter server had been coerced by the adversary, the sensitive data of each client

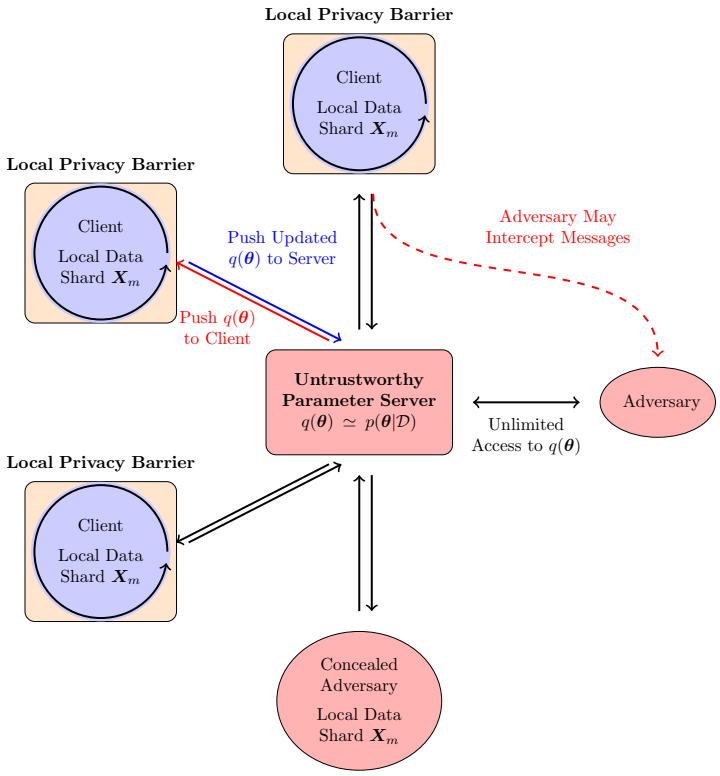


Fig. 3.2 Privacy Barriers assumed by the datapoint level DP-PVI algorithm.

would be protected. Additionally, each client is able to tune individual privacy settings depending on the level of protection desired.

Fig. 3.2 summarises the privacy and trust barriers implied by the data-point level DP-PVI algorithm.

3.2.2 Dataset Level DP-PVI

Recalling the fundamental definition of differential privacy (Definition 2.1.2), an alternative method of constructing DP-PVI is to consider one 'entry' of the dataset as an **entire data shard**, limiting the total contribution of each data shard to the approximate posterior and then including corrupting noise.

Assume that the approximate posterior is parametrised with parameters $\boldsymbol{\lambda} \in \Lambda$ and denote this approximate posterior as $q(\boldsymbol{\theta}|\boldsymbol{\lambda})$. We now present the Dataset Level DP-PVI algorithm, detailed in Algorithm 3.

Noting that the sum of Gaussian random variables is also Gaussian, Equations (3.4) and (3.6) are equivalent to the following update rule:

$$\boldsymbol{\lambda}^{(i)} = \boldsymbol{\lambda}^{(i-1)} + \eta \cdot \left[\sum_{j=1}^{M} \frac{\Delta \boldsymbol{\lambda}_j}{\max(1, ||\Delta \boldsymbol{\lambda}_j||_2)} + \mathcal{N}\left(\mathbf{0}, \sigma^2 C^2\right) \right] \tag{3.7}$$

Algorithm 3 Dataset Level DP-PVI

Input: Partitioned Dataset $\mathcal{D} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_M\}$, Prior $\boldsymbol{\lambda}^{(p)}$, Learning Rate $\eta \in [0, 1]$, Clipping Bound C, DP Noise Scale σ

Initialise approximate likelihoods:

$$t_m^{(0)}(\boldsymbol{\theta}) \leftarrow 1 \ \forall m \tag{3.1}$$

Initialise parameters of approximate posterior:

$$\boldsymbol{\lambda}^{(0)} \leftarrow \boldsymbol{\lambda}^{(p)} \tag{3.2}$$

for $i = 1, 2, \ldots,$ until convergence **do**
  for $j = 1, 2, \ldots, M$ **do** ▷ For each client
    Compute new parameters for this client:

$$\boldsymbol{\lambda}_j \leftarrow \underset{\boldsymbol{\lambda} \in \Lambda}{\operatorname{argmax}} \int q(\boldsymbol{\theta}|\boldsymbol{\lambda}) \ln \frac{q(\boldsymbol{\theta}|\boldsymbol{\lambda}^{(i-1)}) p(\boldsymbol{X}_j|\boldsymbol{\theta})}{q(\boldsymbol{\theta}|\boldsymbol{\lambda}) t_j^{(i-1)}(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \tag{3.3}$$

    $\Delta\boldsymbol{\lambda}_j \leftarrow \boldsymbol{\lambda}_j - \boldsymbol{\lambda}^{(i-1)}$
    Clip and corrupt update:

$$\tilde{\Delta}\boldsymbol{\lambda}_j = \eta \cdot \left[\frac{\Delta\boldsymbol{\lambda}_j}{\max(1, ||\Delta\boldsymbol{\lambda}_j||_2 / C)} + \mathcal{N}\Big(\mathbf{0}, \frac{\sigma^2 C^2}{M}\Big) \right] \tag{3.4}$$

    $\boldsymbol{\lambda}_j \leftarrow \boldsymbol{\lambda}^{(i)} + \tilde{\Delta}\boldsymbol{\lambda}_j$
    Update the approximate likelihood:

$$t_j^{(i)}(\boldsymbol{\theta}) \leftarrow \frac{q(\boldsymbol{\theta}|\boldsymbol{\lambda}_j)}{q(\boldsymbol{\theta}|\boldsymbol{\lambda}^{(i-1)})} t_j^{(i-1)}(\boldsymbol{\theta}) \tag{3.5}$$

  end for
  Compute new global parameters:

$$\boldsymbol{\lambda}^{(i)} = \boldsymbol{\lambda}^{(i-1)} + \sum_{j=1}^{M} \tilde{\Delta}\boldsymbol{\lambda}_j \tag{3.6}$$

  Update privacy cost using the Moments Accountant.
end for

which uses the Gaussian mechanism with subsampling to produce a differentially private estimate of the parameter update and applies a partial update. Unlike in gradient descent methods, the value of the learning rate is meaningful as a full parameter update (neglecting clipping) would give the current best guess of the optimal settings whilst in gradient descent methods, the learning rate controls step sizes without any indication of what step size is correct. Note that the learning rate is in the range $[0, 1]$.

The rationale behind distributing the central noise across each client and performing clipping locally is to ensure that each client is able to accurately track their contribution to the global approximate posterior; if clipping and noise corruption occurred centrally when calculating the new global parameters, the local approximate likelihood terms would become out-of-sync with the global approximate posterior, resulting in sub-optimal parameter values.

A variant on this scheme would instead have each client transmit exact parameter updates to the central parameter server. Clipping and noise corruption would then occur at the server, **which would be required to recalculate the approximate likelihood term for each client**. Whilst the clipping of the update from each client naturally corresponds to an approximate likelihood which the server would have the recalculate, there is freedom in precisely how the noise is incorporated into the client approximate likelihoods.

Note that Eq. (3.4) is also applying the Gaussian mechanism to the update from each client as the clipping fixes the ℓ_2 sensitivity. Recalling Eq. 2.4, it can be seen that for fixed δ, the division of the variance by M weakens the privacy guarantee by a factor of $\sqrt{M}$. Thus for large M, the messages from client to server will effectively be insecure, meaning that this **proposal relies on the existence of secure encryption methods** to prevent the adversary intercepting these messages.

The distribution of noise has additional consequences. Firstly, global parameter updates must be performed in parallel; if updates are performed using messages from one client only, it is possible that other clients will be able to infer sensitive information from the approximate posterior update. This could be problematic when there are an exceeding large number of clients, but a simple remedy would be to perform updates on subsets of clients, fixing the number of clients after which an update will occur. There would be a compromise in the size of such a subset; if the size is small, we effectively add a much quantity larger of noise for each epoch (after all data has been used) but the update from each client will take into account the information from other clients better. Secondly, since each client is aware of the noise they have contributed, they would be able to remove this noise from the global parameter update. Therefore, effectively,from the point of view of a client, the variance of the noise protecting other clients is reduced by a factor of $\frac{M-1}{M}$. For a large number of clients, **assuming that each concealed adversary** is not in collusion, this will have a small effect on the privacy guarantees provided. However, this raises privacy concerns in cases where it is possible that the adversary is able to conceal themselves and place multiple clients in the system. For instance, an incredibly simplified example is a situation where the clients

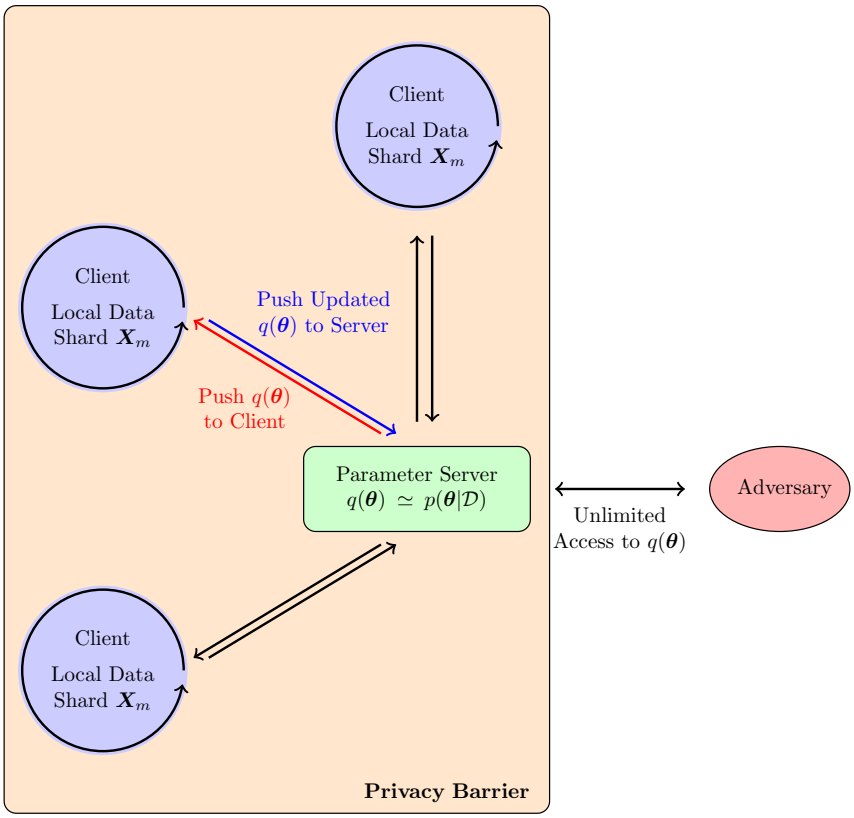


Fig. 3.3 Privacy Barriers assumed by the dataset level DP-PVI algorithm.

consist of one genuine client and a large number of concealed adversaries in collusion. In this case, the adversaries would be able to collude to remove almost all of the noise added by the parameter server rendering the sensitive information of the genuine client insecure.

Additionally, we remark that this techniques requires that each client is able to ensure the authenticity of the parameter server; if the adversary masqueraded as the parameter server, it could inform each client that there are a very large number of clients, after which access to each message will enable the adversary to recover sensitive information about each client.

Fig. 3.3 summaries the trust and privacy barrier assumptions implied by the dataset level DP-PVI algorithm.

3.2.3 Comparison of Dataset and Datapoint Level Protection

As mentioned in the previous sections, there are significant differences between which parties are assumed to be trustworthy between the dataset and data-point level protection schemes. In practice, if choosing between these schemes, this will be a key factor in assessing which scheme is appropriate.

Besides the differences in implied trust and privacy barriers, there is a crucial difference in the type of protection offered by each scheme. The data-point level protection scheme neighbouring datasets to be those which have differing individual data-points whilst the

dataset level allows for differences in an entire data-shard. We now mathematically compare the protection offered by these schemes.

Let $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^N$ and let $\mathcal{D}^{(t)}$ denote a dataset which differs from $\mathcal{D}$ in t entries. Consider the definition of data-point level differential privacy i.e. assume that neighbouring datasets have one value of $\boldsymbol{x}_i$ that differs:

$$\begin{aligned}
\Pr(\mathcal{A}(\mathcal{D}) \in S) &\leq e^{\epsilon}\Pr(\mathcal{A}(\mathcal{D}^{(1)}) \in S) + \delta \\
&\leq e^{2\epsilon}\Pr(\mathcal{A}(\mathcal{D}^{(2)}) \in S) + e^{\epsilon}\delta + \delta \\
&\vdots \\
&\leq e^{t\epsilon}\Pr(\mathcal{A}(\mathcal{D}^{(t)}) \in S) + \delta\left[1 + e^{\epsilon} + e^{2\epsilon} + \ldots + e^{(t-1)\epsilon}\right] \\
&\leq e^{t\epsilon}\Pr(\mathcal{A}(\mathcal{D}^{(t)}) \in S) + \delta\frac{1-e^{t\epsilon}}{1-e^{\epsilon}} \qquad (3.8)
\end{aligned}$$

Therefore, if we protect each data-point with (ϵ, δ) differentially privacy, a group of t data-points is protected with $(t\epsilon, \delta(1-e^{t\epsilon})/(1-e^{\epsilon}))$ differential privacy. The dataset level DP-PVI approach can be regarded as providing an (ϵ, δ) guarantee on any changes within each shard which is itself made up of N data-points. Whilst the above analysis cannot be used to convert the group (ϵ, δ) pair into (ϵ, δ) at the data-point level, we can interpret dataset level (ϵ, δ) protection as being 'stronger' than the equivalent data-point level protection. Intuitively, it ought to be more difficult to disguise changes of an entire data-shard compared to a single data-point, matching the above analysis which shows that data-point level (ϵ, δ) protection corresponds to a much weaker guarantee on the group.

ms2314: I want to discuss the argument made here (above)

Additionally, we remark that in cases where the each data-point in a client's data shard corresponds to an individual, the data-point level differential privacy approach is the more natural way to quantify privacy offered as then the (ϵ, δ) guarantee formed corresponds directly to outcomes being 'similar' for datasets which neighbour in the sense that only the specific information about a single individual is different. Similarly for the case where an entire data shard corresponds to a single user, the data-point level approach seems to be excessive and the dataset level approach is more natural.

ms2314: Also want to discuss the above

Chapter 4

Case Study: Bayesian Linear Regression

Here, both dataset and data-point level DP-PVI is applied to a Gaussian linear regression model.

4.1 Preliminaries

ms2314: Unsure how much detail I ought to give here

4.1.1 Model Definition

Data at each client is generated according to:

$$y_i = \theta x_i + \epsilon_i \qquad \epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma_e^2) \tag{4.1}$$

where θ is a fixed, unknown parameter which is the same for every client. Each client has observations, $\boldsymbol{X}_m = \{(x_i^{(m)}, y_i^{(m)})\}_{i=1}^{N_m} = (\boldsymbol{x}_m, \boldsymbol{y}_m)$. Denote the entire dataset as $\mathcal{D} = \{\boldsymbol{X}_m\}_{m=1}^{M}$. A Gaussian prior is placed on θ:

$$p(\theta) = \mathcal{N}(\mu_\theta, \sigma_\theta^2) \tag{4.2}$$

The approximate likelihood factors take the form:

$$t_i(\theta) \propto \mathcal{N}(\mu_i, \sigma_i^2) \tag{4.3}$$

meaning that the approximate posterior, $q(\theta)$, is a Gaussian distribution. The aim is to find a $q(\theta)$ which is a good approximation to $p(\theta|\mathcal{D})$. Note that this posterior distribution is also a Gaussian distribution.

It is useful to express the univariate Gaussian distribution using *natural parameters*.

$$\begin{aligned}\mathcal{N}(x|\mu,\sigma^2) &= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\Big(-\frac{1}{2\sigma^2}(x-\mu)^2\Big)\\ &= \frac{1}{\sqrt{2\pi}}\exp\Bigg(\underbrace{\begin{bmatrix}1/\sigma^2\\ \mu/\sigma^2\end{bmatrix}}_{\boldsymbol{\eta}}\cdot\underbrace{\begin{bmatrix}-x^2/2\\ x\end{bmatrix}}_{\boldsymbol{T}(x)}-(\frac{\mu^2}{2\sigma^2}-\ln\sigma)\Bigg)\\ &= h\exp[\boldsymbol{\eta}\cdot\boldsymbol{T}(x)-A(\boldsymbol{\eta})] \qquad (4.4)\end{aligned}$$

with

$$A(\boldsymbol{\eta}) = \frac{\eta_2^2}{2\eta_1}-\frac{1}{2}\ln\eta_1 \qquad (4.5)$$

A consequence of this representation is that the product of two Gaussian distributions is that:

$$\mathcal{N}(x|\boldsymbol{\eta}_1)\cdot\mathcal{N}(x|\boldsymbol{\eta}_2) \propto \mathcal{N}(x|\boldsymbol{\eta}_1+\boldsymbol{\eta}_2) \qquad (4.6)$$

$$\mathcal{N}(x|\boldsymbol{\eta}_1)/\mathcal{N}(x|\boldsymbol{\eta}_2) \propto \mathcal{N}(x|\boldsymbol{\eta}_1-\boldsymbol{\eta}_2) \qquad (4.7)$$

which may be seen by direct substitution. Note that the parameter $\eta_1 = 1/\sigma^2$ is known as the *precision*.

4.1.2 Analytical Update Equations

For this model, the update equations given by Equations (2.20) and (2.21) are analytical.

Recall Eq. (2.23) which reformulates the free energy maximisation as a $\mathcal{KL}$ minimisation between q and the tilted distribution. Let j be the client index. This $\mathcal{KL}$ is minimised when $q(\theta)$ is exactly equal to $\hat{p}_j^{(i)}(\theta)$. The tilted distribution takes the form

$$\begin{aligned}\hat{p}_m^{(i)}(\theta) &\propto \frac{q^{(i-1)}(\theta)}{t_m^{(i-1)}(\theta)}p(\boldsymbol{X}_m|\theta)\\ &\propto \underbrace{\mathcal{N}(\theta|\boldsymbol{\eta}_q^{(i-1)}-\boldsymbol{\eta}_m^{(i-1)})}_{\text{'prior'}}\overbrace{p(\boldsymbol{X}_m|\theta)}^{\text{likelihood}} = \mathcal{N}(\theta|\tilde{\boldsymbol{\eta}}) \qquad (4.8)\end{aligned}$$

with

$$\tilde{\eta}_1 = \eta_{q,1}^{(i-1)}-\eta_{m,1}^{(i-1)}+\frac{\boldsymbol{x}_m^T\boldsymbol{x}_m}{\sigma_e^2} \qquad (4.9)$$

$$\tilde{\eta}_2 = \eta_{q,2}^{(i-1)}-\eta_{m,2}^{(i-1)}+\frac{\boldsymbol{x}_m^T\boldsymbol{y}_m}{\sigma_e^2} \qquad (4.10)$$

since this is equivalent to standard Bayesian linear regression, applying the equations for the exact posterior (as found in Bishop [2006]). Thus, the free energy maximisation step is equivalent to setting $\boldsymbol{\eta}_q^{(i)} = \tilde{\boldsymbol{\eta}}$ as defined above.

The natural parameters of the approximate likelihood term are now straightforward to calculate as follows:

$$\boldsymbol{\eta}_m^{(i)} = \boldsymbol{\eta}_q^{(i)} + \boldsymbol{\eta}_m^{(i-1)} - \boldsymbol{\eta}_q^{(i-1)} \tag{4.11}$$

Note that since the approximate posterior, $q(\theta)$, must normalise, there is no need to keep track of the scale factors of the approximate likelihood terms; only the functional dependence upon θ must be stored.

4.1.3 Gradient of Local Free Energy

We now derive the gradient of the local free energy, defined in Eq. (2.22) with respect to the mean and variance of q. For client m, the free energy is written:

$$\begin{aligned}\mathcal{F}_m^{(i)}(q(\theta)) &= \int q(\theta) \ln \frac{q^{(i-1)}(\theta) p(\boldsymbol{X}_m|\theta)}{q(\theta) t_m^{(i-1)}(\theta)} \, d\theta \\ &= \mathcal{H}[q] + \int q(\theta) \ln p(\boldsymbol{X}_m|\theta) \, d\theta + \int q(\theta) \ln \mathcal{C} \cdot \mathcal{N}(\theta|\boldsymbol{\eta}_q^{(i-1)} - \boldsymbol{\eta}_m^{(i-1)}) \, d\theta \end{aligned} \tag{4.12}$$

where $\mathcal{H}[q]$ is the *differential entropy* of q and $\mathcal{C}$ is some constant. The differential entropy for a Gaussian random variable has a simple analytical form:

$$\mathcal{H}[q] = \frac{1}{2}(1 + \ln 2\pi\sigma_q^2) \tag{4.13}$$

[Bishop, 2006] and thus its gradients are straightforward. By substitution, the likelihood term can be written as follows:

$$\begin{aligned}\int q(\theta) \ln p(\boldsymbol{X}_m|\theta) \, d\theta &= \mathcal{C} - \frac{1}{2\sigma_e^2} \int q(\theta) \left\{ \theta^2 \boldsymbol{x}_m^T \boldsymbol{x}_m - 2\theta \boldsymbol{x}_m^T \boldsymbol{y}_m \right\} d\theta \\ &= \mathcal{C} - \frac{1}{2\sigma_e^2} \left\{ (\mu_q^2 + \sigma_q^2) \boldsymbol{x}_m^T \boldsymbol{x}_m - 2\mu_q \boldsymbol{x}_m^T \boldsymbol{y}_m \right\} \end{aligned} \tag{4.14}$$

where the first and second moments of the Gaussian distribution have been directly substituted in. The gradients of this term are straightforward to evaluate. Let $\tilde{\mu}$ and $\tilde{\sigma}^2$ denote the

mean and variance corresponding to $\boldsymbol{\eta}_q^{(i-1)} - \boldsymbol{\eta}_m^{(i-1)}$. Then, the final term can be written as:

$$\begin{aligned}\int q(\theta) \ln \mathcal{C} \cdot \mathcal{N}(\theta | \boldsymbol{\eta}_q^{(i-1)} - \boldsymbol{\eta}_m^{(i-1)})\, d\theta &= \mathcal{C} - \frac{1}{2\tilde{\sigma}^2} \int q(\theta) \{\theta^2 - 2\tilde{\mu}\theta\}\, d\theta \\ &= \mathcal{C} - \frac{1}{2\tilde{\sigma}^2} \{\sigma_q^2 + \mu_q^2 - 2\tilde{\mu}\mu_q\} \end{aligned} \tag{4.15}$$

Combining the above expressions and taking derivatives yields the gradients of the local free energy:

$$\frac{\partial \mathcal{F}_m^{(i)}(q(\theta))}{\partial \mu_q} = -\frac{1}{\sigma_e^2}(\boldsymbol{x}_m^T \boldsymbol{x}_m \mu_q - \boldsymbol{x}_m^T \boldsymbol{y}_m) - \frac{1}{\tilde{\sigma}^2}(\mu_q - \tilde{\mu}) \tag{4.16}$$

$$\frac{\partial \mathcal{F}_m^{(i)}(q(\theta))}{\partial \sigma_q^2} = \frac{1}{2\sigma_q^2} - \frac{1}{\sigma_e^2}(\boldsymbol{x}_m^T \boldsymbol{x}_m) - \frac{1}{\tilde{\sigma}^2} \tag{4.17}$$

4.1.4 Assessing Performance

ms2314: We use KL blah blah blah to assess performance, is this good? no, comment on this etc

4.2 Datapoint Level DP-PVI

ms2314: Need to explain how the data is generated

4.2.1 DP-SGA

With the gradient of the free energy calculated in the previous section, Algorithm 1 can be adapted to perform gradient ascent on the local free energy. Recalling Algorithm 2 and differentially private stochastic gradient ascent thus yields an algorithm which is differentially private which respect to the data-points over each data shard.

In our implementation, gradient ascent is performed on the mean and the log of the variance in order to ensure that variance remains positive; otherwise, large learning rates can give negative variances, after which gradient calculations are meaningless and the algorithm fails. Writing $\hat{\sigma}^2 = \ln \sigma_q^2$, the gradient of the free energy for the log of the variance is written:

$$\frac{\partial \mathcal{F}_m^{(i)}(q(\theta))}{\partial \hat{\sigma}_q^2} = \frac{\partial \mathcal{F}_m^{(i)}(q(\theta))}{\partial \sigma_q^2} \cdot \exp(\hat{\sigma}_q^2) \tag{4.18}$$

Additionally, after private gradients for the mean and log-variance are calculated, if there magnitude exceeds a set threshold, their magnitude is reduced to this threshold value.

This avoids problems with relatively learning rates causing instability during the course of training.

Fig. 4.1 outlines typical results for this approach produced by hand tuning hyper-parameters. We remark that the form of gradient clipping involved in the DP-SGA is problematic as there is no guarantee that the scale of gradients of different parameters is similar but clipping is performed with regards to the ℓ_2 norm and the noise applied is isotropic. It is reassuring to see that this approach yields $\mathcal{KL}$ divergences which are small (10^{-3}), showing that we are able to achieve performance effectively equivalent to the non-private approach but the privacy cost is quite high, requiring $\epsilon \simeq 500$ for reasonable performance.

Fig. 4.2 plots the privacy cost assuming a Gaussian mechanism with a varying sampling probability and shows there is a very large dependence of ϵ upon q. This figure suggests that such an approach can only give strong privacy protection when there are a large number of data-points within each data shard which would enable a small value of q to also give a reliable gradient estimate. This is particularly difficult in the federated learning context as the data is distributed across clients, but will be appropriate in certain cases.

4.2.2 Analytical Updates

An alternative approach to perform the local free energy maximisation is to use adapt the exact analytical equations to form a differentially private mechanism. Equations (4.9) and (4.10) can be modified as follows:

$$\ell_{m,n} = \sqrt{(x_{m,n}^2)^2 + (x_{m,n}y_{m,n})^2} \tag{4.19}$$

$$\tilde{\eta}_1 = \eta_{q,1}^{(i-1)} - \eta_{m,1}^{(i-1)} + \frac{1}{\sigma_e^2} \max\left\{0, \left[\sum_{n=1}^{N_m} \frac{x_{m,n}^2}{\max(1, \ell_{m,n}/C)} + \mathcal{N}(0, C^2\sigma^2)\right]\right\} \tag{4.20}$$

$$\tilde{\eta}_2 = \eta_{q,2}^{(i-1)} - \eta_{m,2}^{(i-1)} + \frac{1}{\sigma_e^2}\left[\sum_{n=1}^{N_m} \frac{x_{m,n}y_{m,n}}{\max(1, \ell_{m,n}/C)} + \mathcal{N}(0, C^2\sigma^2)\right] \tag{4.21}$$

which bounds the ℓ_2 sensitivity by using clipping and applies the Gaussian mechanism with noise scaled according to the gradient clipping bound C. Note that a mechanism employing sub-sampling is not appropriate here as each application of the above mechanism needs to estimate the true maximum of the free energy. In Eq. 4.20, an additional clipping step is applied to ensure the contribution from the corrupted term remains positive (otherwise the precision may become negative which causes numerical problems).

Additionally, it is worth nothing that the above update equations provide unbiased estimates

ms2314: estimates is probably not the right word for this

for the *natural parameters* of the tilted posterior, but this does not correspond to unbiased estimates of the mean and variance of the title posterior as seen in Fig. 4.3 showing that,

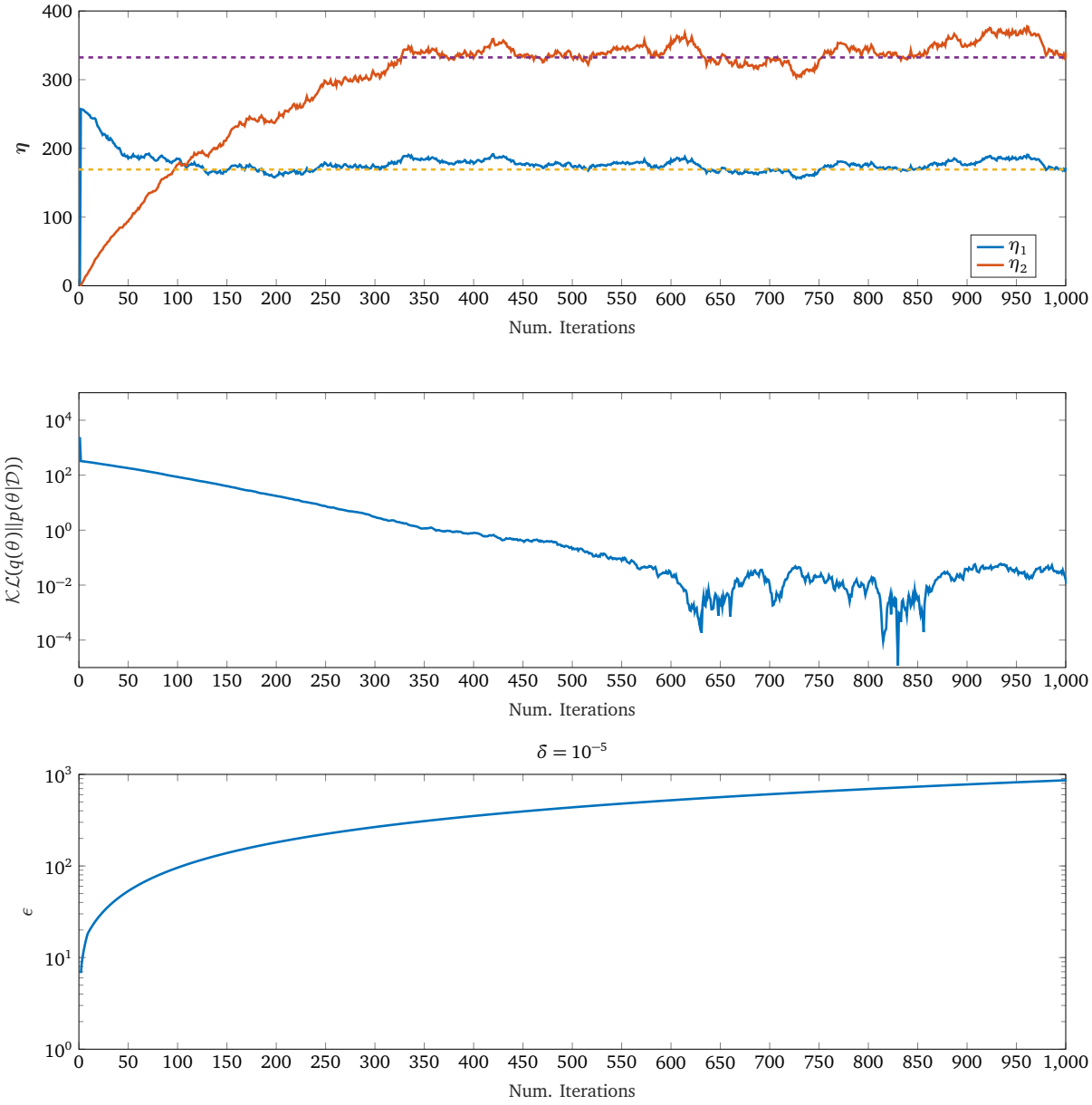


Fig. 4.1 Differentially Private SGA Results. First figure outlines evolution of the natural parameters stored at the central parameter server as the number of iterations increases. The $\mathcal{KL}$ divergence between $q(\theta)$ and the true posterior is also plotted (middle sub-plot), as is privacy guarantee, ϵ, with $\delta = 10^{-5}$ fixed (bottom sub-plot). Parameters used: $\theta = 2$, $\sigma_e = 0.5$, $\mu_\theta = 0$, $\sigma_\theta = 5$, $\eta = 10^{-5}$, $L = 1$, $C = 10$, $\sigma = 1$. 5 workers with 10 points per worker. 50 iterations of DP-SGA performed locally for one server update.

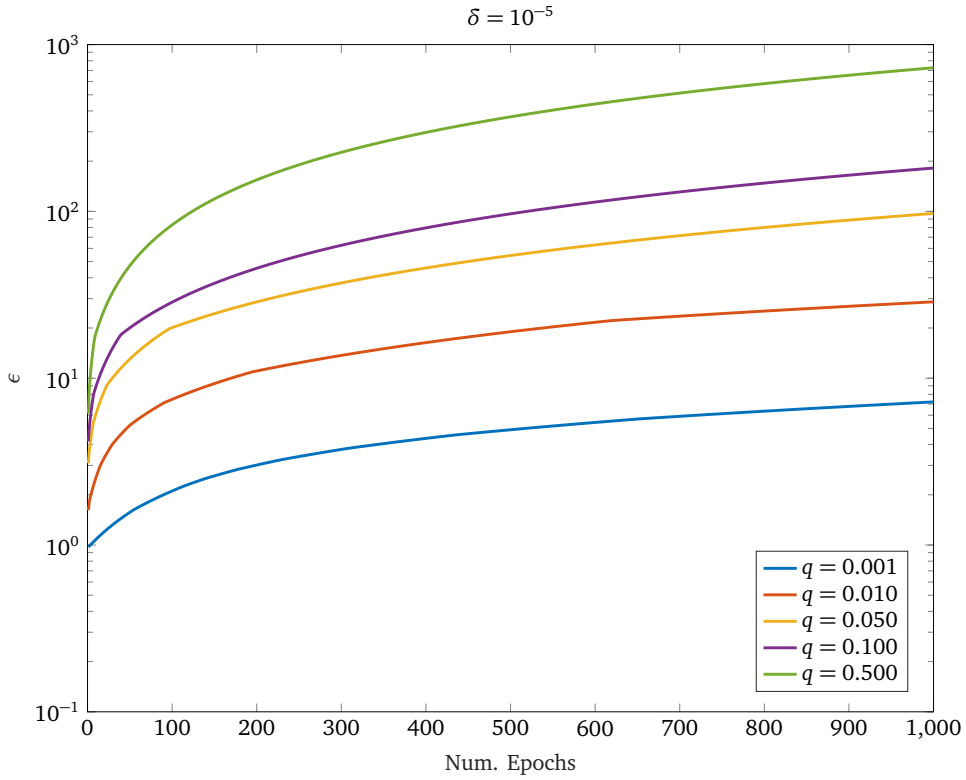


Fig. 4.2 Privacy cost for different sampling probabilities, $q = L/N$, as a function of the number of epochs assuming a mechanism which sub-samples from each dataset. An epoch is defined as the mechanism having been run processed $\frac{1}{q}$ times i.e. N data-points having been visited.

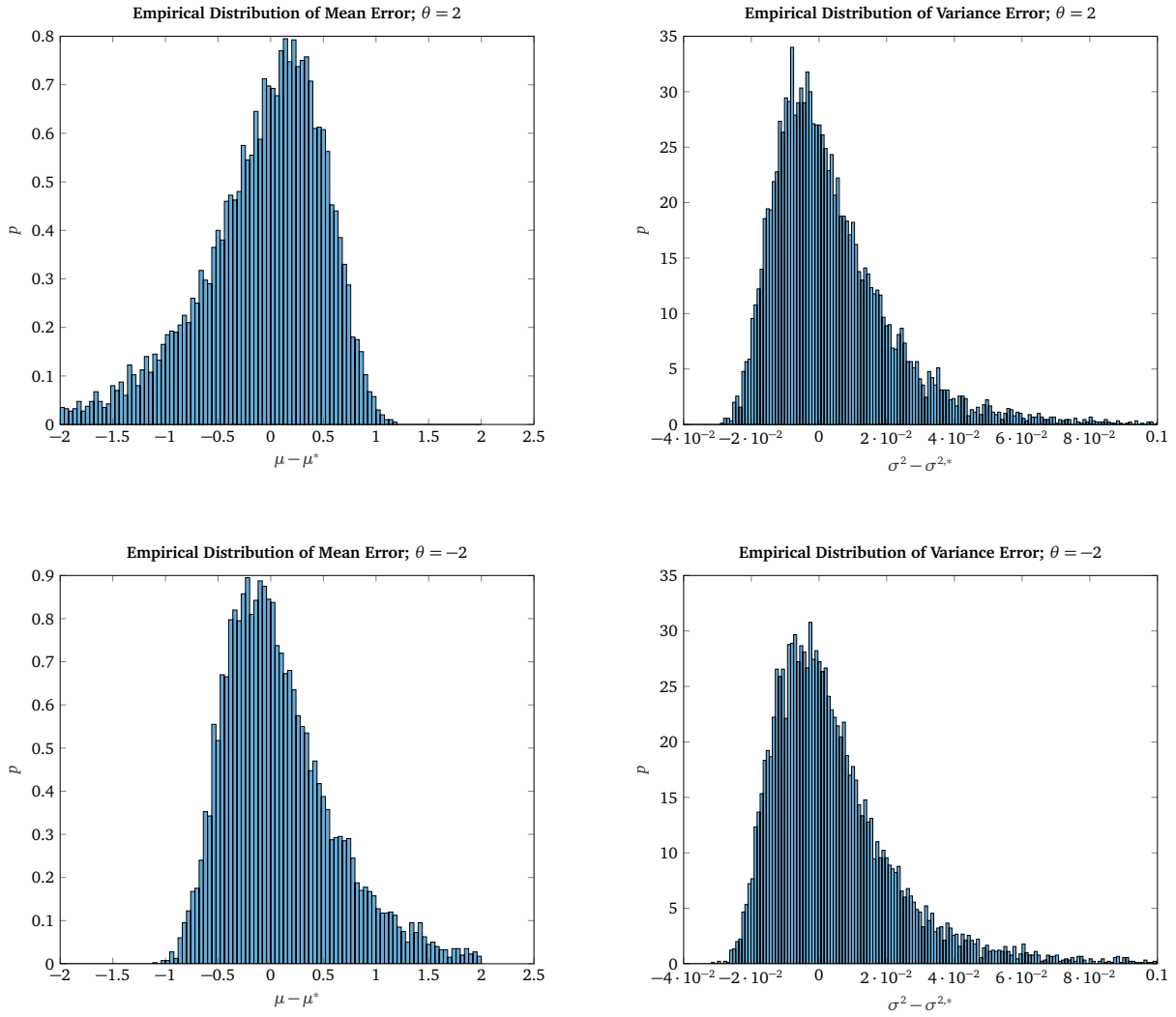


Fig. 4.3 Simulation on introduced bias on titled posterior calculation. Prior, $p(\theta) = \mathcal{N}(0, 5)$ with $\sigma_e = 0.5$, 10 data-points spaced uniformly between in $[-1, 1]$. $N = 10000$ draws of differential privacy corrupting noise (with $\sigma = 1$) used to produce empirical distributions. Clipped neglected.

assuming no clipping occurs, the absolute value of the mean is slightly underestimated and the variance tends to be overestimated.

Since *a priori*, it is not known that the true ranges of each value of x and y will be, the choice of clipping bound, C, is crucial. If chosen too small, the solution obtained by each maximisation is incorrect and bias is introduced into the results. However, if chosen too large, the solution will be dominated by noise. This issue could be avoided if it were possible to rescale the range and mean of numerical data, but this would also have to be done privately and is complicated by the fact that the data is distributed across client.

Fig. 4.4 shows results obtained using this approach. It is immediately clear that increasing the clipping bound value tends to increase the variance of the $\mathcal{KL}$ divergence, likely due to the standard deviation of the corrupting noise scaling with the clipping bound. We also note that these standard deviations are very high and inspecting the bottom left figure, there are

very large differences in performance for the same parameter settings. The median being significantly lower than the mean for all values of C shows that performance can be incredibly poor for certain datasets, which is far from optimal. Curiously, the best median performance achieved corresponds to $C = 0.25$, the smallest value used, which gives a median $\mathcal{KL}$ of 22. Fig. 4.5 shows private posteriors and true posteriors corresponding to this level of $\mathcal{KL}$ divergence, showing reasonable performance. It is worth noting that the private posteriors underestimate the precision of the true posterior and have a mean value closer to 0, almost acting as a sort of regularisation. This method does perform worse than the DP-SGA approach (which achieves effectively the non-private solution), but achieves significantly better privacy costs. Additionally, we note that increasing the value of the clipping bound tends to decrease performance significantly as it corresponds to adding significantly larger values of noise.

ms2314: Need to think/discuss this argument in more detail

The value of θ has a large effect on the performance; smaller values of theta tend to give much lower values of $\mathcal{KL}$ divergence whilst increasing the absolute value of θ gives worse performance. This can be justified as follows: firstly, noting the form of the $\mathcal{KL}$ divergence, there is a large penalty when $p(\theta|D)$ is small and $q(\theta)$ is not, corresponding to severe penalties on differences in the mean parameter. Secondly, assuming fixed values of $\boldsymbol{x}$ and ϵ, changing θ directly affects η_2 (the product of the mean and precision of the posterior) but does not affect η_1. Therefore a larger value of θ corresponds to a larger value of η_1 and thus a larger value of $\ell_{m,n}$. For moderate values of clipping bound, this results in larger values of θ giving smaller values of η_1 as can be seen (Fig. 4.4, top left) which are then relatively more noisy as the applied noise is isotropic. Since the mean is obtained as η_1/η_2. Since the mean is calculated as η_2/η_1, this results in very large fluctuations of the mean value for large η_2, and thus very large fluctuations of the $\mathcal{KL}$ which is very sensitive to this value. When θ is small, the results mean value is also small and less sensitive to the exact value of the precision, giving this behaviour. Noting that for the largest values of θ, smaller values of C correspond to better KL divergences provides evidence that it is fluctuations in the precision giving rise to poor KL divergences as the smallest values of C correspond to the smallest values of additive noise.

Indeed, increasing C appears to increase the ratio between the average precision across the last 10 iterations and the true precision, suggesting that this approach gives a systematic bias. Recalling Eq. 4.20, the cause of this bias can be seen. Assuming that $C > \ell_{m,n}\ \forall(n)$,

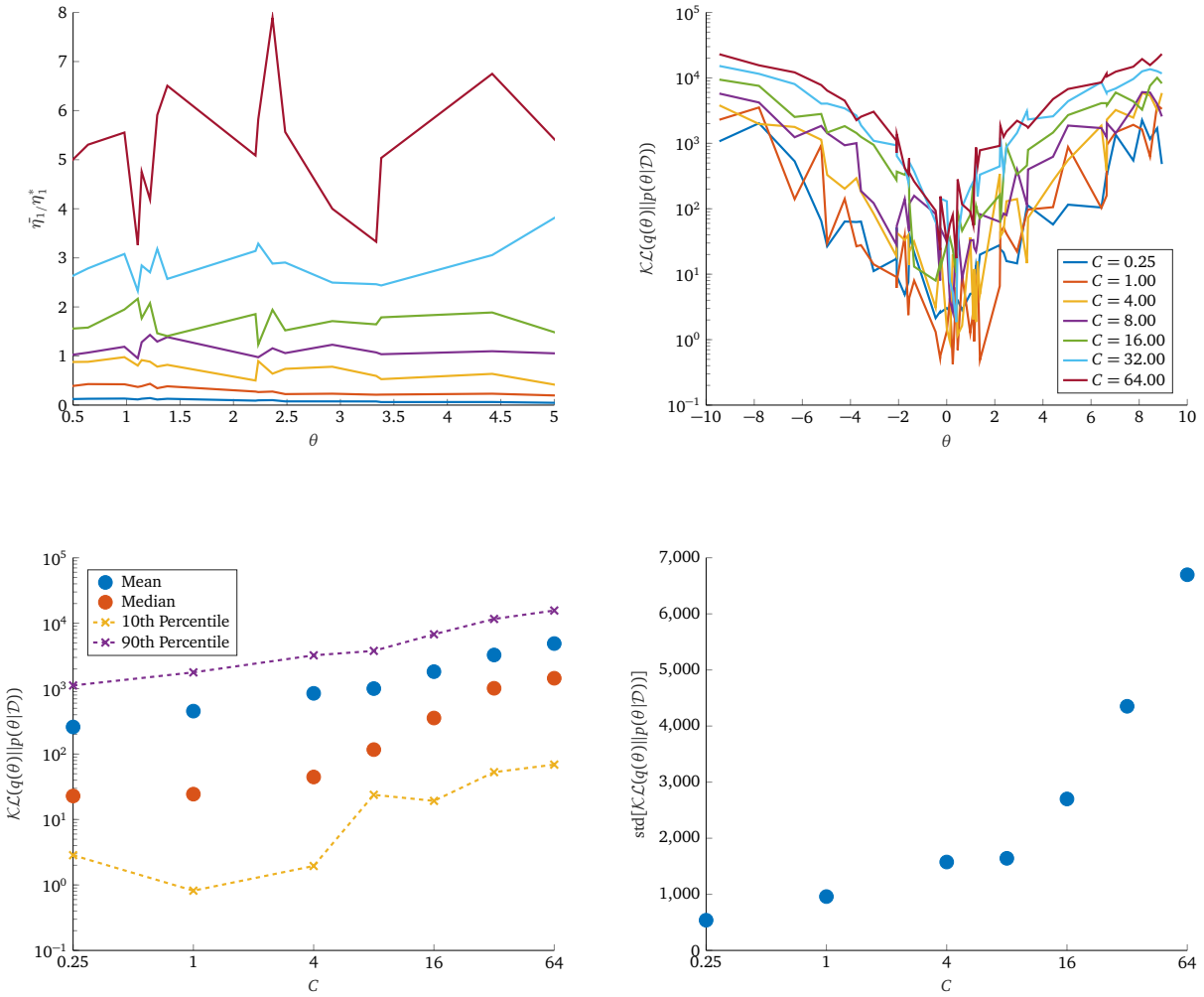


Fig. 4.4 Results obtained with data-point level differential privacy with $\epsilon_{\max} = 10$ (i.e. termination after ϵ exceeds this value). Top left plot shows ratio between the true precision and obtained precision as a function of θ, averaged over the last ten iterations of the DP-PVI algorithm. Top right plot shows the $\mathcal{KL}(q||p)$ (again averaged over the ten last iterations) as a function of θ. Bottom left plots show the mean, median and chosen percentiles of this averaged $\mathcal{KL}$ divergence as a function of C (across the 50 different values of θ) and the bottom right plot shows the corresponding standard deviation. 50 random seeds used for each clipping bound value with each random seed corresponding to a specific θ sampled from the prior distribution, $p(\theta) = \mathcal{N}(0, 5)$ and specific draw of corrupting noise. $\sigma_e = 0.5$, $\sigma = 5$, $\eta = 0.1$ with $M = 20$ workers and 10 data-points per worker.

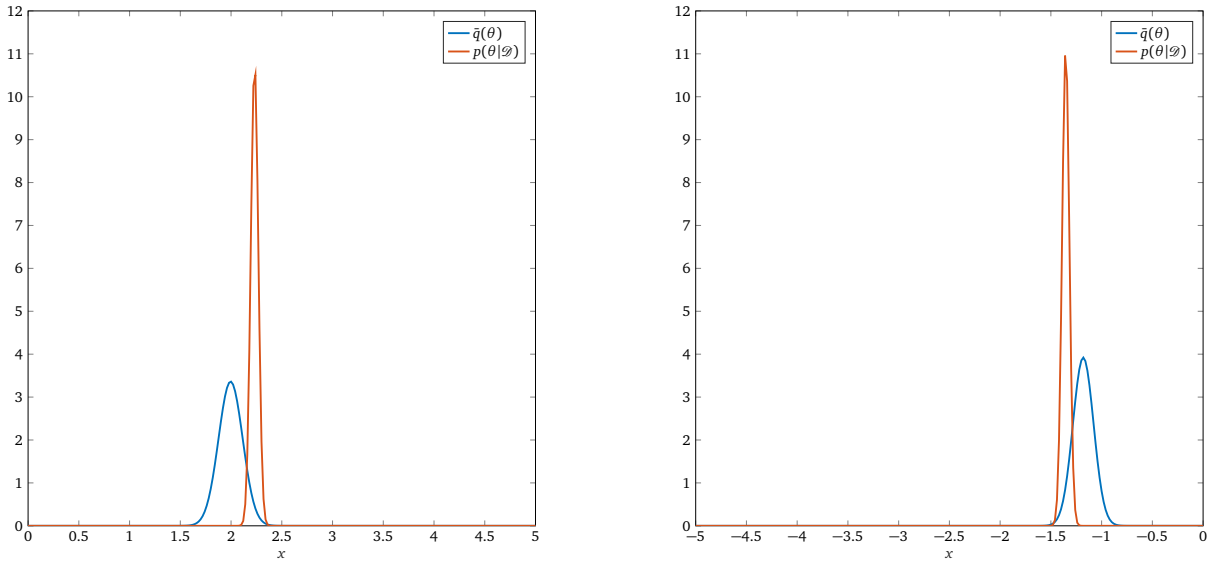


Fig. 4.5 Typical posteriors obtained using the data-point level DP-PVI algorithm with analytical clipped updates. Produced with $C = 2$, $\eta = 0.1$, $M = 20$ workers with 10 data-points per worker.

the update can be written as:

$$\tilde{\eta}_1 = \eta_{q,1}^{(i-1)} - \eta_{m,1}^{(i-1)} + \frac{1}{\sigma_e^2} \underbrace{\max\left\{0, \left[\overbrace{\sum_{n=1}^{N_m} x_{m,n}^2}^{\Sigma_{xx}} + \mathcal{N}(0, C^2\sigma^2)\right]\right\}}_{\Delta}$$

$$= \eta_{q,1}^{(i-1)} - \eta_{m,1}^{(i-1)} + \frac{1}{\sigma_e^2}\Delta \tag{4.22}$$

There would be no systematic bias if $\mathbb{E}[\Delta] = \Sigma_{xx}$. Taking $\boldsymbol{X}_m$ to be fixed:

$$\Delta = \max\{0, \mathcal{N}(\Sigma_{xx}, C^2\sigma^2)\} \tag{4.23}$$

and thus the distribution of Δ can be written by inspection as follows:

$$p(\Delta = x) = \Phi\left(\frac{-\Sigma_{xx}}{\sigma C}\right)\delta(x) + \begin{cases} 0 & x \leq 0 \\ \mathcal{N}(x|\Sigma_{xx}, \sigma^2C^2) & x > 0 \end{cases} \tag{4.24}$$

where $\Phi(\cdot)$ denotes the cumulative density function of the standard Gaussian distribution. This distribution clearly has mean larger than Σ_{xx} as negative contributions in the first moment from are replaced with a zero contribution due to the delta function. As the value of C increases, the the negative contribution which is removed increases in value, resulting in a larger bias, which matches the obtained results.

ms2314: I could do the analytical maths for this and plot if, if I had time

Fig. 4.6 shows results for the same parameter settings as Fig. 4.4, confirming that it is not the clipping which introduces bias in the precision estimate but rather bias introduced by the noise; the top left figure shows that the precision is only ever underestimated. As expected, increasing the clipping bound increases model performance when no noise is applied. Fig. 4.7 shows typical approximate posteriors obtained for the extremes of the clipping bound; large C gives practically unmodified posteriors as expected whilst small values of C show the previously observed pattern of underestimating the precision and absolute value of θ.

Whilst further gains in terms of reduction of the privacy expenditure could likely be made by a more comprehensive search over hyper-parameters, this scheme does not give particularly good performance in practice for this model. However, it may be practical to use this in certain contexts with a small clipping bound in order to avoid the overestimation of the precision. It is then likely that the approximate posterior formed will have a larger variance and smaller mean (in terms of absolute value) compared to the true posterior, but since the bias appears to be systematic, the machine learning practitioner may be able to correct for this or at the minimum be aware of the consequences.

4.2.3 Hydrid Scheme

Whilst the DP-SGA scheme is able to provide performance indistinguishable from the non-private performance, it does so at relatively high privacy costs. On the other hand, analytical updates provide an approximate, heavily unbiased solution but at low privacy costs. These approaches could be combined straightforwardly, using the analytical update to initialise the DP-SGA scheme in a suitable position, which would then require fewer iterations to reach the optimum solution.

It is likely that this would achieve better privacy bounds than the DP-SGA approach at significantly lower privacy costs, though it is unclear exactly how large the privacy gains would be. It is suggested that this approach, which could be applied in general for exponential conjugate models, should be investigated further, but it is not investigated further in this report.

4.3 Dataset Level DP-PVI

Algorithm 3 can applied to this probabilistic model directly using the analytical update equations derived previously. In our implementation, the approximate posterior is parametrised using it's natural parameters and thus the clipping and corruption step (Eq. 3.4) clipping and corrupts changes in the precision and the product of the mean and precision directly.

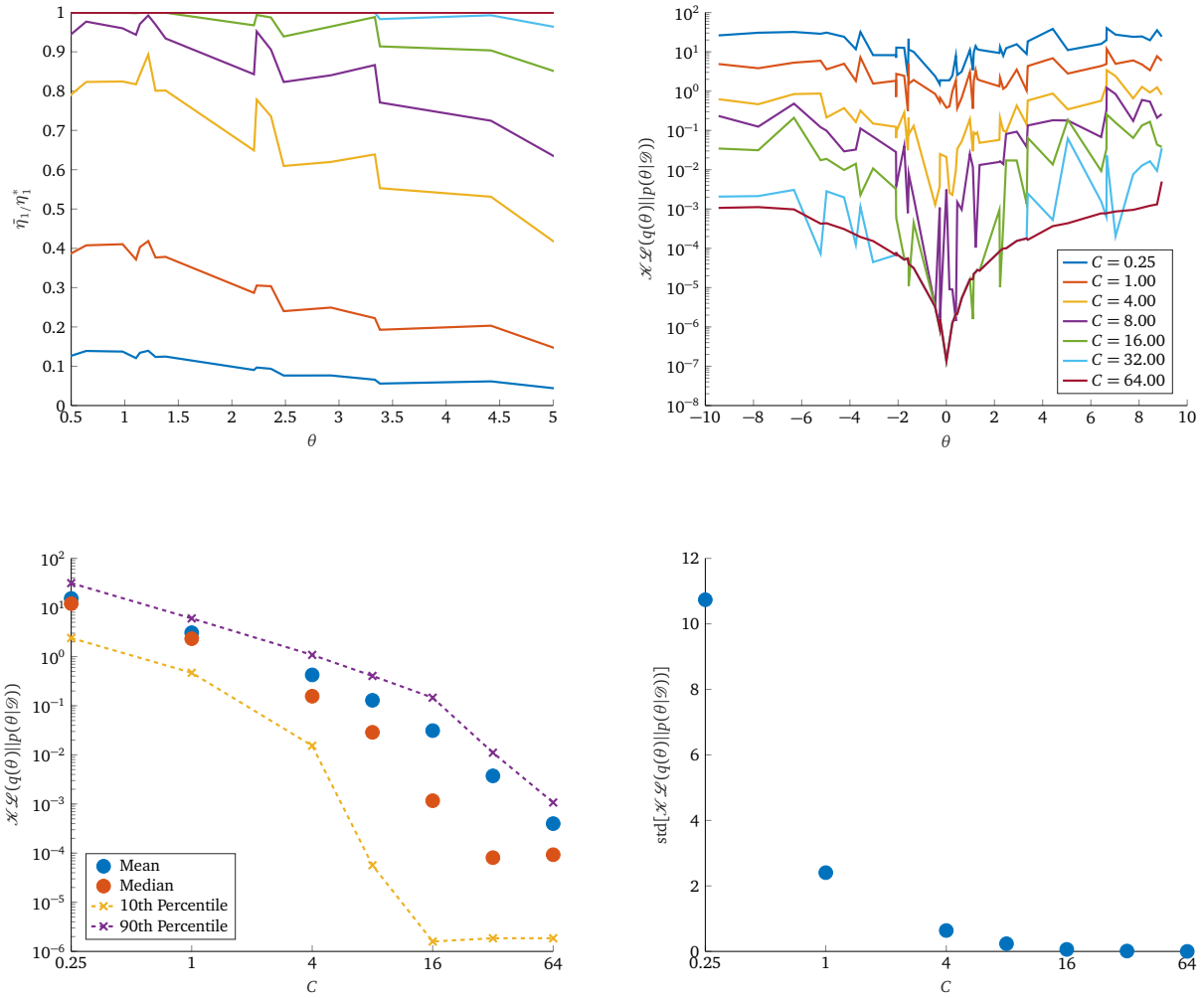


Fig. 4.6 Results obtained with data-point level differential privacy applied **without noise** when run for the same number of iterations as Fig. 4.4. Top left plot shows ratio between the true precision and obtained precision as a function of θ, averaged over the last ten iterations of the DP-PVI algorithm. Top right plot shows the $\mathcal{KL}(q||p)$ (again averaged over the ten last iterations) as a function of θ. Bottom left plots show the mean, median and chosen percentiles of this averaged $\mathcal{KL}$ divergence as a function of C (across the 50 different values of θ) and the bottom right plot shows the corresponding standard deviation. 50 random seeds used for each clipping bound value with each random seed corresponding to a specific θ sampled from the prior distribution, $p(\theta) = \mathcal{N}(0, 5)$ and specific draw of corrupting noise. $\sigma_e = 0.5$, $\eta = 0.1$ with $M = 20$ workers and 10 data-points per worker.

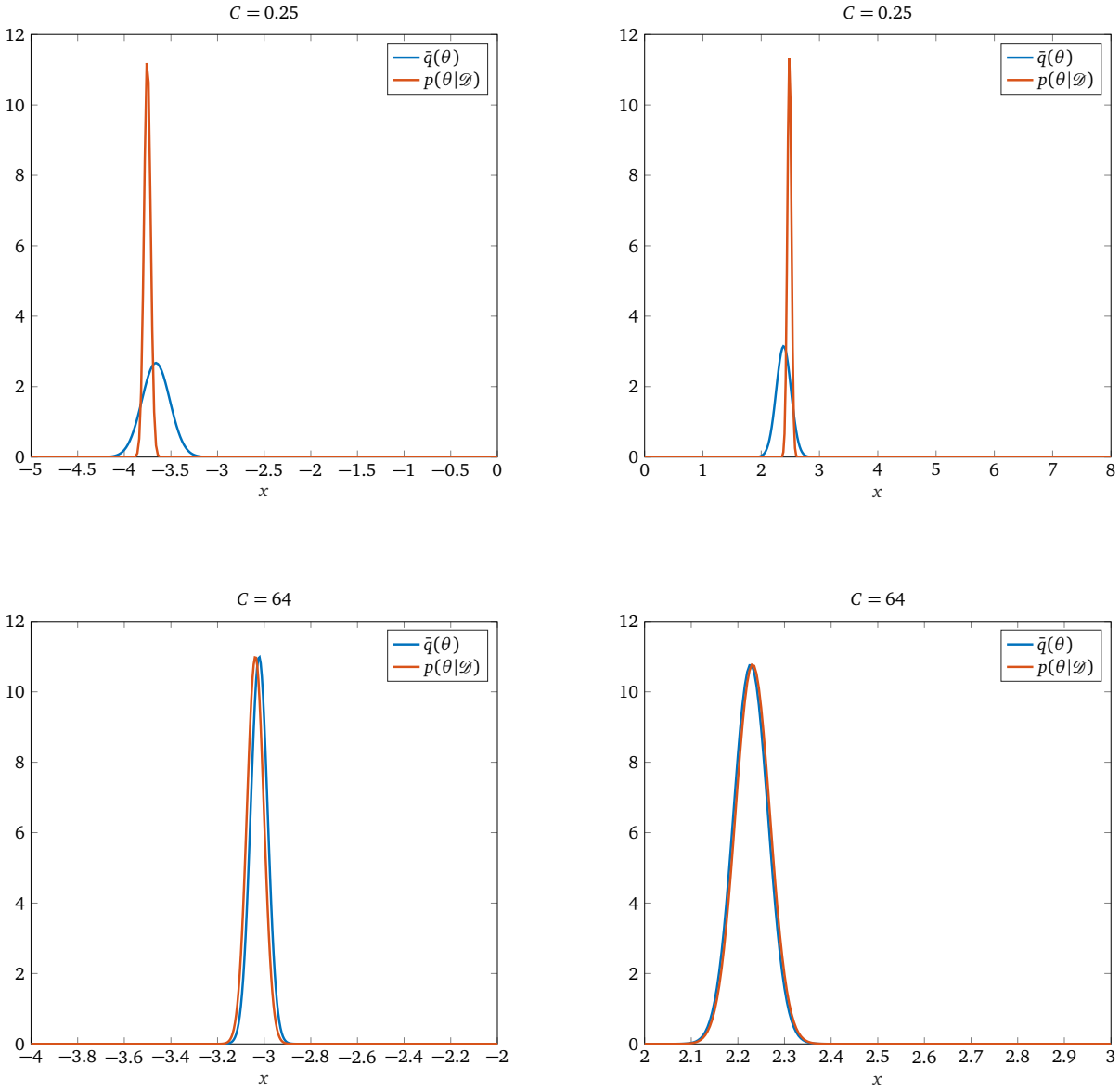


Fig. 4.7 Typical posteriors obtained using the **noiseless** data-point level DP-PVI algorithm with analytical clipped updates. Produced with $C = 2$, $\eta = 0.1$, $M = 20$ workers with 10 data-points per worker.

Chapter 5

Conclusions

References

Teppo Niinimäki, Mikko Heikkilä, Antti Honkela, and Samuel Kaski. Representation transfer for differentially private drug sensitivity prediction. *arXiv preprint arXiv:1901.10227*, 2019.

A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125, May 2008. doi: 10.1109/SP.2008.33.

Melissa Gymrek, Amy L McGuire, David Golan, Eran Halperin, and Yaniv Erlich. Identifying personal genomes by surname inference. *Science*, 339(6117):321–324, 2013.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 1322–1333, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3832-5. doi: 10.1145/2810103.2813677. URL http://doi.acm.org/10.1145/2810103.2813677.

Federated learning: Collaborative machine learning without centralized training data, Apr 2017. URL https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, August 2014. ISSN 1551-305X. doi: 10.1561/0400000042. URL http://dx.doi.org/10.1561/0400000042.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 308–318, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4139-4. doi: 10.1145/2976749.2978318. URL http://doi.acm.org/10.1145/2976749.2978318.

J. Jälkö, O. Dikmen, and A. Honkela. Differentially Private Variational Inference for Non-conjugate Models. *ArXiv e-prints*, October 2016.

Thang D Bui, Cuong V Nguyen, Siddharth Swaroop, and Richard E Turner. Partitioned variational inference: A unified framework encompassing federated and continual learning. *arXiv preprint arXiv:1811.11206*, 2018.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. ISBN 978-0387-31073-2. URL http://research.microsoft.com/en-us/um/people/cmbishop/prml/.

Appendix A

Risk Assessment Retrospective

In risk assessment submitted it October, it was noted that the risks related to this project would be due to extensive computer usage. This risk assessment proved to be accurate and the appropriate steps were taken to mitigate these risks, including ensuring appropriate working conditions, suitable posture and a sensible working pattern.

Appendix B

Electronic Resources

Please note that all code listings for this project may be found on `GitHub` at this address: https://github.com/MrinankSharma/dp-pvi-project.

Additionally, the electronic log book used for this project can be found at the following address: https://drive.google.com/file/d/1PafemIYA0pPCeIDv2K_utI8CYklXyPmM/view?usp=sharing.