Master's thesis
Master's Programme in Data Science

Differentially Private Markov Chain Monte Carlo

Ossi Räisä

January 20, 2021

Supervisor(s): Associate Professor Antti Honkela

Examiner(s): Associate Professor Antti Honkela
Dr. Antti Koskela

University of Helsinki
Faculty of Science

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Degree programme
Faculty of Science	Master's Programme in Data Science

Tekijä — Författare — Author

Ossi Räisä

Työn nimi — Arbetets titel — Title

Differentially Private Markov Chain Monte Carlo

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidantal — Number of pages
Master's thesis	January 20, 2021	44

Tiivistelmä — Referat — Abstract

ACM Computing Classification System (CCS):

Avainsanat — Nyckelord — Keywords

Differential Privacy, Markov Chain Monte Carlo

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Contents

1. Introduction

2. Background

2.1 Differential Privacy

Differential privacy [DR14] is a property of an algorithm that quantifies the amount of information about private data an adversary can gain from the publication of the algorithm's output. The most commonly used definition uses two real numbers, ϵ and δ, to quantify the information gain, or, from the perspective of a data subject, the privacy loss of the algorithm.

The most common definition is called (ϵ, δ)-DP, approximate DP or ADP [DR14]. The case where $\delta = 0$ is called ϵ-DP or pure DP.

Definition 1. *An algorithm $\mathcal{M}\colon \mathcal{X} \to \mathcal{U}$ is (ϵ, δ)-ADP if for all neighbouring inputs $x \in \mathcal{X}$ and $x' \in \mathcal{X}$ and all measurable sets $S \subset \mathcal{U}$*

$$P(\mathcal{M}(x) \in S) \leq e^{\epsilon} P(\mathcal{M}(x') \in S) + \delta.$$

The neighbourhood relation in the definition is domain specific. With tabular data the most common definitions are the add/remove neighbourhood and substitute neighbourhood.

Definition 2. *Two tabular datasets are said to be add/remove neighbours if they are equal after adding or removing at most one row to or from one of them. The datasets are said to be in substitute neighbours if they are equal after changing at most one row in one of them.*

The neighbourhood relation is denoted by $\sim$. The definitions and theorems of this section are valid for all neighbourhood relations.

There many other definitions of differential privacy that are mostly used to compute (ϵ, δ)-bounds for ADP. This thesis uses two of them: Rényi-DP (RDP) [Mir17] and zero-concentrated differential privacy (zCDP) [BS16]. Both are based on Rényi divergence [Mir17], which is a particular way of measuring the difference between random variables.

Definition 3. *For random variables with density or probability mass functions P and Q the Rényi divergence of order $1 < \alpha < \infty$ is*

$$D_\alpha(P \,||\, Q) = \frac{1}{\alpha - 1} \ln E_{x\sim Q}\left(\frac{P(x)^\alpha}{Q(y)^\alpha}\right).$$

Orders $\alpha = 1$ and $\alpha = \infty$ are defined by continuity:

$$D_1(P \,||\, Q) = \lim_{\alpha\to 1-} D_\alpha(P \,||\, Q),$$

$$D_\infty(P \,||\, Q) = \lim_{\alpha\to \infty} D_\alpha(P \,||\, Q).$$

Both Rényi-DP and zCDP can be expressed as bounds on the Rényi divergence between the outputs of an algorithm with neighbouring inputs:

Definition 4. *An algorithm $\mathcal{M}$ is (α, ϵ)-Rényi DP if for all $x \sim x'$*

$$D_\alpha(\mathcal{M}(x) \,||\, \mathcal{M}(x')) \leq \epsilon.$$

$\mathcal{M}$ is ρ-zCDP if for all $\alpha > 1$ and all $x \sim x'$

$$D_\alpha(\mathcal{M}(x) \,||\, \mathcal{M}(x')) \leq \rho\alpha.$$

Rényi-DP and zCDP bounds can be converted to ADP bounds [Mir17, BS16]:

Theorem 1. *If $\mathcal{M}$ is (α, ϵ)-RDP, $\mathcal{M}$ is also $(\epsilon - \frac{\ln\delta}{\alpha-1}, \delta)$-ADP for any $0 < \delta < 1$. If $\mathcal{M}$ is ρ-zCDP, $\mathcal{M}$ is also $(\rho + \sqrt{-4\rho\ln\delta}, \delta)$-ADP for any $0 < \delta < 1$.*

A very useful property of all of these definitions is composition [DR14]: if algorithms $\mathcal{M}$ and $\mathcal{M}'$ are DP, the algorithm first computing $\mathcal{M}$ and then $\mathcal{M}'$, outputting both results, is also DP, although with worse bounds. More precisely

Definition 5. *Let $\mathcal{M}\colon \mathcal{X} \to \mathcal{U}$ and $\mathcal{M}'\colon \mathcal{X}\times\mathcal{U} \to \mathcal{U}'$ be algorithms. Their composition is the algorithm outputting $(\mathcal{M}(x), \mathcal{M}'(x, \mathcal{M}(x)))$ for input x.*

Theorem 2. *Let $\mathcal{M}\colon \mathcal{X} \to \mathcal{U}$ and $\mathcal{M}\colon \mathcal{X}\times\mathcal{U} \to \mathcal{U}'$ be algorithms. Then*

1. *If $\mathcal{M}$ is (ϵ, δ)-ADP and $\mathcal{M}'$ is (ϵ', δ')-ADP, then their composition is $(\epsilon+\epsilon', \delta+\delta')$-ADP [DR14]*
2. *If $\mathcal{M}$ is (α, ϵ)-RDP and $\mathcal{M}'$ is (α, ϵ')-RDP, then their composition is $(\alpha, \epsilon+\epsilon')$-RDP [Mir17]*
3. *If $\mathcal{M}$ is ρ-zCDP and $\mathcal{M}'$ is ρ'-zCDP, then their composition is $(\rho+\rho')$-zCDP [BS16]*

All of the composition results can be extended to any number of compositions by induction. Note that any step of the composition can depend on the results of the previous steps, not only on the private data. There are also other composition theorems for ADP that trade increased δ for decreased ϵ or vice-versa, but this thesis does not apply them directly.

As any algorithm that does not use private data in any way is $(0, 0)$-ADP, 0-zCDP and $(\alpha, 0)$-RDP with all α, Theorem 2 has the following corollary, called post-processing immunity:

Theorem 3. *Let $\mathcal{M}\colon \mathcal{X} \to \mathcal{U}$ be an ADP, RDP or zCDP algorithm with some privacy parameters. Let $f\colon \mathcal{U} \to \mathcal{U}'$ be any algorithm not using the private data. Then the composition of $\mathcal{M}$ and f is ADP, RDP or zCDP with the same privacy parameters.*

There are many different DP algorithms that are commonly used, which are also called mechanisms [DR14]. This thesis only requires one of the most commonly used ones: the Gaussian mechanism [DR14].

Definition 6. *The Gaussian mechanism with parameter σ^2 is an algorithm that, with input x, outputs a sample from $\mathcal{N}(x, \sigma^2)$, where $\mathcal{N}$ denotes the normal distribution.*

The RDP and zCDP bounds for the Gaussian mechanism are quite simple. The ADP bound is more complicated:

Theorem 4. *If for all inputs x and x', $||x - x'||_2 \leq \Delta$, the Gaussian mechanism is*

1. *$(\alpha, \frac{\alpha\Delta^2}{2\sigma^2})$-RDP [Mir17]*
2. *$\frac{\Delta^2}{2\sigma^2}$-zCDP [BS16]*
3. *n compositions of the Gaussian mechanism are $(\epsilon, \delta(\epsilon))$-ADP [SMM19] with*

$$\delta(\epsilon) = \frac{1}{2}\left(\operatorname{erfc}\left(\frac{\sigma(\epsilon - n\mu)}{\sqrt{2n}\Delta}\right) - e^{\epsilon}\operatorname{erfc}\left(\frac{\sigma(\epsilon + n\mu)}{\sqrt{2n}\Delta}\right)\right),$$

where $\mu = \frac{\Delta^2}{2\sigma^2}$ and erfc *is the complementary error function.*

The most common use case for the Gaussian mechanism is computing a function $f\colon \mathcal{X} \to \mathbb{R}$ of private data and feeding the result into the Gaussian mechanism to privately release the function value. The condition that the inputs of the Gaussian mechanism cannot vary too much leads into the concept of sensitivity of a function

Definition 7. *The l_p-sensitivity Δ_p, with neighbourhood relation $\sim$, of a function $f\colon \mathcal{X} \to \mathbb{R}^n$ is*

$$\Delta_p f = \sup_{x \sim x'} ||f(x) - f(x')||_p.$$

Theorem 4 implies that the value of any function with finite l_2-sensitivity can be privately released using the Gaussian mechanism with appropriate noise variance σ^2. Of course, the usefulness of the released value depends on the magnitude of σ^2 compared to the actual value.

2.2 Bayesian Inference and Markov Chain Monte Carlo

In Bayesian inference, the parameters of a statistical model are inferred from observed data using Bayes' theorem [GCS+14]. The result is not just a point estimate of the parameters, but a probability distribution describing the likelihood of different values of the parameters.

Bayes' theorem relates the *posterior* belief of the parameters $p(\theta \mid X)$ to the *prior* belief $p(\theta)$ through the observed data X and the likelihood of the data $p(X \mid \theta)$ as follows:

$$p(\theta \mid X) = \frac{p(X \mid \theta)p(\theta)}{\int p(X \mid \theta)p(\theta)d\theta}.$$

It is theoretically possible to compute $p(\theta \mid X)$ given any likelihood, prior and data, but the integral in the denominator is in many cases difficult to compute [GCS+14]. In such cases the posterior cannot be feasibly computed. However, many of the commonly used summary statistics of the posterior, such as the mean, variance and credible intervals, can be approximated from a sample of the posterior. *Markov chain Monte Carlo* (MCMC) is a widely used algorithm to obtain such samples [GCS+14].

MCMC algorithms sequentially sample values of θ with the goal of eventually having the chain of sampled values converge to a given distribution [GCS+14]. While this can be done in many ways, this thesis focuses on a particular MCMC algorithm: *Metropolis-Hastings* (MH).

At each iteration i, the Metropolis-Hastings algorithm samples θ_i from a distribution π of the parameters by first picking a proposal θ' from a proposal distribution $q(\theta_{i-1})$ [GCS+14], where θ_{i-1} is the previously sampled value*. We shorten θ_{i-1} to θ in the following. The ratio of posterior and proposal densities is calculated

$$r(\theta, \theta') = \frac{\pi(\theta')}{\pi(\theta)} \frac{q(\theta \mid \theta')}{q(\theta' \mid \theta)},$$

and the proposal is accepted with probability $\min\{1, r\}$. If the proposal is accepted, $\theta_i = \theta$, otherwise $\theta_i = \theta$.

*The value of θ_0 for the first iteration is given as input to the algorithm.

It can be shown that, with a suitable proposal distribution, the chain of θ_i values converges to π [GCS+14]. The Gaussian distribution centered at the current value is a commonly used proposal.

When MCMC is used in Bayesian inference, the distribution to approximate is

$$\pi(\theta) = p(\theta \mid X) = \frac{p(X \mid \theta)p(\theta)}{\int p(X \mid \theta)p(\theta)d\theta}.$$

The difficult integral $\int p(X \mid \theta)p(\theta)d\theta$ in the denominator cancels out when computing r, so only the likelihood and the prior are needed. For numerical stability, r is usually computed in log space, which makes the acceptance probability $\min\{1, e^{\lambda(\theta,\theta')}\}$ where

$$\lambda(\theta, \theta') = \ln \frac{p(X \mid \theta')}{p(X \mid \theta)} + \ln \frac{p(\theta')}{p(\theta)} + \ln \frac{q(\theta \mid \theta')}{q(\theta' \mid \theta)}.$$

The dataset X is typically a table with n independent rows. The likelihood is given as $p(x_j \mid \theta)$ for row x_j. Independence of the rows means that

$$p(X \mid \theta) = \prod_{j=1}^{n} p(x_j \mid \theta)$$

which means that the log likelihood ratio term of λ is

$$\ln \frac{p(X \mid \theta')}{p(X \mid \theta)} = \sum_{j=1}^{n} \ln \frac{p(x_j \mid \theta')}{p(x_j \mid \theta)}$$

Algorithm 1 puts all of this together to summarise the MH algorithm used for Bayesian inference.

Algorithm 1: Metropolis-Hastings: number of iterations k, proposal distribution q and initial value θ_0 and dataset X as input

for $1 \leq i \leq k$ **do**

denote $\theta = \theta_{i-1}$

sample $\theta' \sim q(\theta)$

$\ln p(X \mid \theta) = \sum_{j=1}^{n} (\ln p(x_j \mid \theta') - \ln p(x_j \mid \theta))$

$\lambda = \ln p(X \mid \theta) + \ln p(\theta') - \ln p(\theta) + \ln q(\theta \mid \theta') - \ln q(\theta' \mid \theta)$

$$\theta_i = \begin{cases} \theta' & \text{with probability } \min\{1, e^{\lambda}\} \\ \theta & \text{otherwise} \end{cases}$$

end

return $(\theta_1, \ldots, \theta_k)$

2.3 The Banana Distribution

The banana distribution [TPK14] is a banana-shaped probability distribution that is a challenging target for MCMC algorithms. For this reason it has been used to test MCMC algorithms in the literature [TPK14].

Definition 8. *Let X have a d-variate Gaussian distribution with mean μ and covariance matrix Σ. Let*

$$g(x) = (x_1, x_2 - a(x_1 - m)^2 - b, x_3, \ldots, x_d),$$

with $a, b, m \in \mathbb{R}$. The banana distribution with parameters μ, Σ, a, b and m is the distribution of $g(X)$. It is denoted by $\mathrm{Ban}(\mu, \Sigma, a, b, m)$.

In the literature, the banana distribution is simply used as the target to sample from, and is not the posterior in a Bayesian inference problem [TPK14]. To test differentially private MCMC algorithms, the target distribution must be the posterior of some inference problem, as otherwise there is no data to protect with differential privacy. Theorem 5 gives a suitable inference problem for testing DP MCMC algorithms. Figure 2.1 shows the posterior of a 2-dimensional model.

Theorem 5. *Let*

$$\begin{aligned}
\theta = (\theta_1, \ldots, \theta_d) &\sim \mathrm{Ban}(0, \sigma_0^2 I, a, b, m) \\
X_1 &\sim \mathcal{N}(\theta_1, \sigma_1^2) \\
X_2 &\sim \mathcal{N}(\theta_2 + a(\theta_1 - m)^2 + b, \sigma_2^2) \\
X_3 &\sim \mathcal{N}(\theta_3, \sigma_3^2) \\
&\vdots \\
X_d &\sim \mathcal{N}(\theta_d, \sigma_d^2)
\end{aligned}$$

Given data $x_1, \ldots, x_d \in \mathbb{R}^n$ and denoting $\tau_i = \frac{1}{\sigma_i^2}$, the posterior of θ tempered with T is the banana distribution $\mathrm{Ban}(\mu, \Sigma, a, b, m)$ *with*

$$\bar{x}_i = \frac{1}{n} \sum_{j=1}^{n} x_{ji} \quad i \in \{1, 2\}$$

$$\mu = \left(\frac{Tn\tau_1 \bar{x}_1}{Tn\tau_1 + \tau_0}, \ldots, \frac{Tn\tau_d \bar{x}_d}{Tn\tau_d + \tau_0} \right),$$
$$\Sigma = \mathrm{diag}\left(\frac{1}{Tn\tau_1 + \tau_0}, \ldots, \frac{1}{Tn\tau_d + \tau_0} \right).$$

Proof. Because

$$g^{-1}(y) = (y_1, y_2 + a(y_1 - m)^2 + b, y_3 \ldots, y_d)$$

and the Jacobian determinant of g^{-1} is 1, for a positive-definite Σ the banana distribution has density proportional to

$$\exp\left(-\frac{1}{2} (g^{-1}(x) - \mu)^T \Sigma^{-1} (g^{-1}(x) - \mu) \right)$$

With $\Sigma = \text{diag}(\sigma_1^2, \sigma_2^2)$ the density is proportional to

$$\exp\left(-\frac{1}{2}\left(\left(\frac{x_1-\mu_1}{\sigma_1}\right)^2 + \left(\frac{x_2 + a(x_1-m)^2 + b - \mu_2}{\sigma_2}\right)^2 + \sum_{i=3}^{d}\left(\frac{x_i-\mu_i}{\sigma_i}\right)^2\right)\right)$$

Denote $u = \theta_2 + a(\theta_1 - m)^2 + b$. The tempered posterior of θ is

$$\begin{aligned}
p(\theta \mid X) &\propto p(X \mid \theta)^T p(\theta) \\
&= p(X_1 \mid \theta_1)^T p(X_2 \mid \theta_1, \theta_2)^T \prod_{i=3}^{d} p(X_i \mid \theta_i)^T p(\theta) \\
&= p(X_1 \mid \theta_1)^T p(X_2 \mid \theta_1, \theta_2)^T \prod_{i=3}^{d} p(X_i \mid \theta_i)^T \\
&\cdot \exp\left(-\frac{1}{2}\left(\tau_0\theta_1^2 + \tau_0(\theta_2 + a(\theta_1 - m)^2 + b)^2 \sum_{i=3}^{d} \tau_0\theta_i^2\right)\right) \\
&= p(X_1 \mid \theta_1)^T p(X_2 \mid \theta_1, \theta_2)^T \exp\left(-\frac{1}{2}\left(\tau_0\theta_1^2 + \tau_0(\theta_2 + a(\theta_1 - m)^2 + b)^2\right)\right) \\
&\cdot \prod_{i=3}^{d} p(X_i \mid \theta_i)^T \exp\left(-\frac{1}{2}\sum_{i=3}^{d}\tau_0\theta_i^2\right)
\end{aligned}$$

Considering the upper and lower part of the last expression separately

$$
\begin{aligned}
&p(X_1 \mid \theta_1)^T p(X_2 \mid \theta_1, \theta_2)^T \exp\left(-\frac{1}{2}\left(\tau_0\theta_1^2 + \tau_0(\theta_2 + a(\theta_1 - m)^2 + b)^2\right)\right) \\
&\propto \left(\prod_{i=1}^{n} \exp\left(-\frac{(x_{i1} - \theta_1)^2\tau_1}{2}\right)\right)^T \cdot \left(\prod_{i=1}^{n} \exp\left(-\frac{(x_{i2} - \theta_2 - a(\theta_1 - m)^2 - b)^2\tau_2}{2}\right)\right)^T \\
&\cdot \exp\left(-\frac{1}{2}\left(\tau_0\theta_1^2 + \tau_0(\theta_2 + a(\theta_1 - m)^2 + b)^2\right)\right) \\
&= \exp\left(-\frac{1}{2}\Big(T\tau_1 \sum_{i=1}^{n}(x_{i1} - \theta_1)^2 + T\tau_2 \sum_{i=1}^{n}(x_{i2} - u)^2 + \tau_0\theta_1^2 + \tau_0 u^2\Big)\right) \\
&= \exp\left(-\frac{1}{2}\Big(T\tau_1 \sum_{i=1}^{n}(x_{i1} - \bar{x}_1)^2 + T\tau_1 n(\bar{x}_1 - \theta_1)^2\right. \\
&\left. + T\tau_2 \sum_{i=1}^{n}(x_{i2} - \bar{x}_2)^2 + T\tau_2 n(\bar{x}_2 - u)^2 + \tau_0\theta_1^2 + \tau_0 u^2\Big)\right) \\
&\propto \exp\left(-\frac{1}{2}\Big(T\tau_1 n(\bar{x}_1 - \theta_1)^2 + T\tau_2 n(\bar{x}_2 - u)^2 + \tau_0\theta_1^2 + \tau_0 u^2\Big)\right) \\
&= \exp\left(-\frac{1}{2}\Big(T\tau_1 n\bar{x}_1^2 - 2T\tau_1 n\bar{x}_1\theta_1 + nT\tau_1\theta_1^2 + \tau_0\theta_1^2\right. \\
&\left. + T\tau_2 n\bar{x}_2^2 - 2T\tau_2 n\bar{x}_2 u + nT\tau_2 u^2 + \tau_0 u^2\Big)\right) \\
&\propto \exp\left(-\frac{1}{2}\Big((Tn\tau_1 + \tau_0)\theta_1^2 - 2T\tau_1 n\bar{x}_1\theta_1 + (Tn\tau_2 + \tau_0)u^2 - 2T\tau_2 n\bar{x}_2 u\Big)\right) \\
&= \exp\left(-\frac{1}{2}\Big((Tn\tau_1 + \tau_0)\left(\theta_1^2 - \frac{2T\tau_1 n\bar{x}_1\theta_1}{Tn\tau_1 + \tau_0}\right) + (Tn\tau_2 + \tau_0)\left(u^2 - \frac{2T\tau_2 n\bar{x}_2 u}{Tn\tau_2 + \tau_0}\right)\Big)\right) \\
&\propto \exp\left(-\frac{1}{2}\Big((Tn\tau_1 + \tau_0)\left(\theta_1 - \frac{T\tau_1 n\bar{x}_1}{Tn\tau_1 + \tau_0}\right)^2 + (Tn\tau_2 + \tau_0)\left(u - \frac{T\tau_2 n\bar{x}_2}{Tn\tau_2 + \tau_0}\right)^2\Big)\right)
\end{aligned}
$$

and

$$\begin{aligned}
&\prod_{i=3}^{d} p(X_i \mid \theta_i)^T \cdot \exp\left(-\frac{1}{2}\sum_{i=3}^{d}\tau_0\theta_i^2\right) \\
&\propto \exp\left(-\frac{1}{2}T\sum_{j=3}^{d}\tau_j\sum_{i=1}^{n}(x_{ij}-\theta_j)^2 - \frac{1}{2}\sum_{j=3}^{d}\tau_0\theta_j^2\right) \\
&= \exp\left(-\frac{1}{2}\sum_{j=3}^{d}\left(T\tau_j\sum_{i=1}^{n}(x_{ij}-\theta_j)^2 + \tau_0\theta_j^2\right)\right) \\
&= \exp\left(-\frac{1}{2}\sum_{j=3}^{d}\left(T\tau_j\sum_{i=1}^{n}(x_{ij}-\bar{x}_j)^2 + T\tau_j n(\bar{x}_j-\theta_j)^2 + \tau_0\theta_j^2\right)\right) \\
&\propto \exp\left(-\frac{1}{2}\sum_{j=3}^{d}\left(T\tau_j n(\bar{x}_j-\theta_j)^2 + \tau_0\theta_j^2\right)\right) \\
&\propto \exp\left(-\frac{1}{2}\sum_{j=3}^{d}\left(-2T\tau_j n\bar{x}_j\theta_j + T\tau_j n\theta_j^2 + \tau_0\theta_j^2\right)\right) \\
&\propto \exp\left(-\frac{1}{2}\sum_{j=3}^{d}\left((Tn\tau_j+\tau_0)\theta_j^2 - 2Tn\tau_j\bar{x}_j\theta_j + \frac{(Tn\tau_j\bar{x}_j)^2}{Tn\tau_j+\tau_0}\right)\right) \\
&= \exp\left(-\frac{1}{2}\sum_{j=3}^{d}(Tn\tau_j+\tau_0)\left(\theta_j - \frac{Tn\tau_1\bar{x}_i}{Tn\tau_j+\tau_0}\right)^2\right)
\end{aligned}$$

Multiplying the resulting expression above gives a density proportional to the banana distribution. As $p(\theta \mid X)$ is proportional to the density of a banana distribution, the posterior is the banana distribution $\mathrm{Ban}(\mu, \Sigma, a, b, m)$ with

$$\mu = \left(\frac{Tn\tau_1\bar{x}_1}{Tn\tau_1+\tau_0}, \ldots, \frac{Tn\tau_d\bar{x}_d}{Tn\tau_d+\tau_0}\right),$$

$$\Sigma = \mathrm{diag}\left(\frac{1}{Tn\tau_1+\tau_0}, \ldots, \frac{1}{Tn\tau_d+\tau_0}\right).$$

□

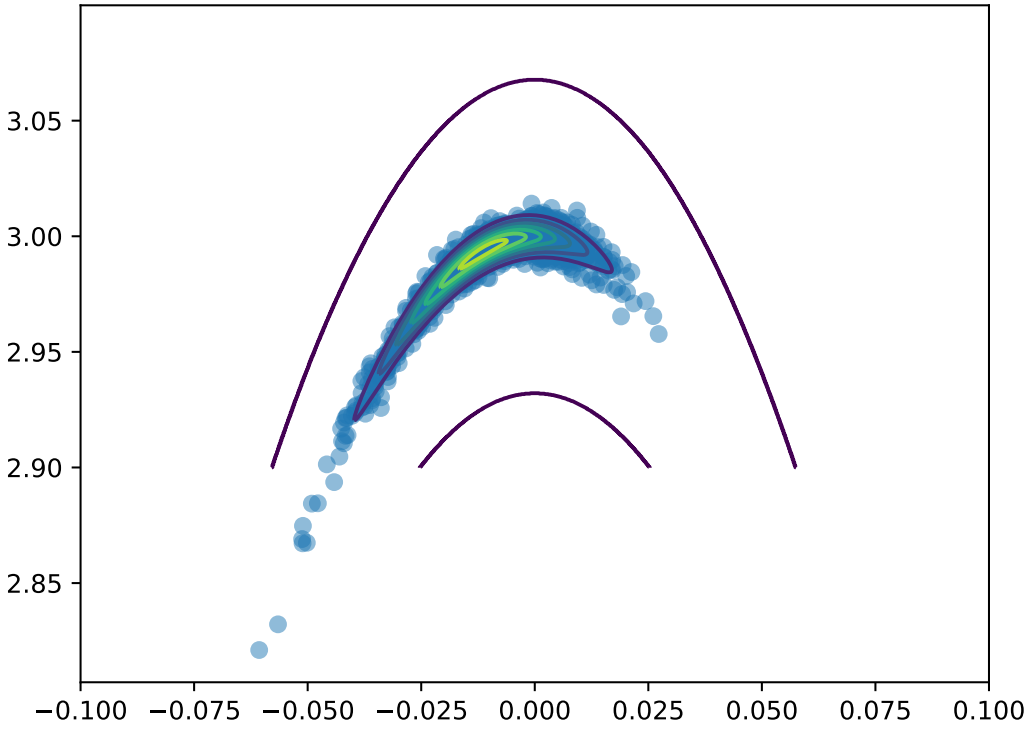


Figure 2.1: Density of the banana model posterior with $a = 20$, $b = m = 0$, $\sigma_1^2 = 20$, $\sigma_2^2 = 2.5$ and $\sigma_0^2 = 1000$. The blue points are 1000 samples from the posterior.

3. Differentially Private MCMC

As seen in Section 2.1, an algorithm can be made differentially private by adding Gaussian noise the its output. The noise could also be added to any intermediate value calculated by the algorithm, and post processing immunity will guarantee that the same DP bounds that hold for releasing the intermediate value also hold for releasing the final result of the algorithm.

In 2019, Yildirim and Ermis [YE19] realised that if Gaussian noise is added to the exact value of λ, the noise can be corrected for yielding a differentially private MCMC algorithm which converges to the correct distribution. In the same year, Heikkilä et. al. [HJDH19] developed another DP MCMC algorithm, called DP Barker, which uses subsampling to amplify privacy.

3.1 DP Penalty

In 1999, Ceperley and Dewing [CD99] developed a variant of Metropolis-Hastings called the penalty algorithm, where only a noisy approximation of λ is known. They developed the algorithm for simulations in physics where computing λ requires computing energies of complex systems, which can only be approximated. The penalty algorithm modifies the acceptance probability to account for the noise added to λ and still converges to the correct distribution if the noise is Gaussian with known variance.

The DP penalty algorithm adds Gaussian noise to the value of λ, and uses the penalty algorithm to correct the acceptance probability so that the algorithm still converges to the correct distribution [YE19]. The corrected acceptance probability for Gaussian noise with variance σ^2 is

$$\min\{1, e^{\lambda(\theta,\theta')-\frac{1}{2}\sigma^2}\}$$

Theorem 6 gives the number of iterations DP penalty can be run for when the privacy cost is computed through zCDP, which is what Yildirim and Ermis prove in their paper [YE19]. A tighter, but harder to use, bound can be reached without using zCDP. This is given by Theorem 7.

Theorem 6. *Let $\epsilon > 0$, $0 < \delta < 1$, $\alpha > 0$ and $\tau > 0$. Let*

$$\rho = (\sqrt{\epsilon - \ln \delta} - \sqrt{-\ln \delta})^2$$

$$c(\theta, \theta') = \sup_{x_j, x_j'} (p(x_j \mid \theta') - p(x_j \mid \theta) - (p(x_j' \mid \theta') - p(x_j' \mid \theta)))$$

$$\sigma^2(\theta, \theta') = \tau^2 n^{2\alpha} c^2(\theta, \theta')$$

Then DP penalty can be run for

$$k = \lfloor 2\tau^2 n^{2\alpha} \rho \rfloor$$

iterations when using σ^2 as the variance of the Gaussian noise.

Theorem 7. *Let $\epsilon > 0$ and $\tau > 0$. Define c and σ as in Theorem 6. The DP penalty algorithm, after running for k iterations using σ as the noise variance, is $(\epsilon, \delta(\epsilon))$-DP for*

$$\delta(\epsilon) = \frac{1}{2} \left(\operatorname{erfc}\left(\frac{\epsilon - k\mu}{2\sqrt{k\mu}} \right) - e^{\epsilon} \operatorname{erfc}\left(\frac{\epsilon + k\mu}{2\sqrt{k\mu}} \right) \right)$$

where $\mu = \frac{1}{2\tau^2 n^{2\alpha}}$.

Proof. DP penalty is an adaptive composition of Gaussian mechanisms that release noisy values of $\lambda(\theta, \theta')$. The sensitivity of $\lambda(\theta, \theta')$ is $c(\theta, \theta')$. For the tight ADP bound used here, the sensitivity must be constant in each iteration. This is achieved by releasing $\frac{\lambda(\theta,\theta')}{c(\theta,\theta')}$ instead, which has sensitivity 1. $c(\theta, \theta')$ does not depend on X, so $\lambda(\theta, \theta')$ can be obtained from $\frac{\lambda(\theta,\theta')}{c(\theta,\theta')}$ by post processing.

Adding Gaussian noise with variance σ_n^2 to $\frac{\lambda(\theta,\theta')}{c(\theta,\theta')}$ is equivalent to adding Gaussian noise with variance $\sigma_n^2 c^2(\theta, \theta')$ to $\lambda(\theta, \theta')$. Setting $\sigma_n^2 = \tau^2 n^{2\alpha}$ and plugging into the ADP bound of Theorem 4 proves the claim. □

Theorem 7 is harder to use than Theorem 6 because the number of iteration DP penalty can be run for given an (ϵ, δ)-bound cannot be computed analytically for the former. However, the maximum number of iterations can be solved for numerically.

Theorems 6 and 7 require a bound on sensitivity of the log likelihood ratio. If there is a bound

$$|\ln p(x_j \mid \theta') - \ln p(x_j \mid \theta)| \leq L||\theta - \theta'||_2$$

for all D_j, θ and θ' then

$$c(\theta, \theta') \leq 2L||\theta - \theta'||_2$$

The former bound is true in some model, such as logistic regression. In other models it can be forced by clipping the log likelihood ratios to the interval $[-L||\theta - \theta'||_2, L||\theta - \theta'||_2]$. This will remove the guarantee of eventually converging to the correct posterior,

but if L is chosen to be large enough, the clipping will not affect the acceptance decision frequently. As a tradeoff, picking a large L will increase the variance of the Gaussian noise and slow down convergence through it.

Yildirim and Ermis [YE19] propose two potential ways to improve the performance of the penalty algorithm. The first improvement is only proposing changes in one dimension in a multidimensional problem. This decreases $||\theta - \theta'||_2$, which means that it decreases the noise variance.

The second improvement is called *guided random walk* (GRW) [YE19]. In GRW, proposals change only one dimension, as above. Additionally, a direction is associated with each dimension, and proposals are only made the current direction of the chosen dimension. After an accepted proposal, the direction is kept the same, but after a reject it is switched. This means that the chain can move towards areas of higher probability faster because, after some initial proposals are rejected, the directions for each dimension point towards the area of high probability, so all proposals are towards it. Without GRW, most proposals would move the chain away from the area of high probability, and would likely be rejected.

3.2 DP Barker

The DP Barker algorithm of Heikkilä et. al. [HJDH19] is based on the Barker acceptance test [Bar65] instead of the Metropolis-Hastings test. Instead of using the MH acceptance probability, the Barker acceptance test samples $V_{log} \sim \text{Logistic}(0, 1)$ and accepts if

$$\lambda + V_{log} > 0$$

If Gaussian noise with variance σ^2 is added to λ, there exists a correction distribution V_{corr} such that $\mathcal{N}(0, \sigma^2) + V_{corr}$ has the same distribution as V_{log}. Because the variance of V_{log} is $\frac{\pi^2}{3}$ [HJDH19], the variance of V_{corr} must be $\frac{\pi^2}{3} - \sigma^2$ which means that there is an upper bound to the noise variance: $\sigma^2 < \frac{\pi^2}{3}$. Testing whether $\lambda + \mathcal{N}(0, \sigma^2) + V_{corr} > 0$ is equivalent to testing whether $\lambda + V_{log} > 0$, which means that it is possible to derive a DP MCMC algorithm based on the Barker acceptance test if the correction distribution can be sampled from.

However, the analytical form of V_{corr} is not known [HJDH19]. Heikkilä et. al. approximate the distribution with a Gaussian mixture model. This means that their algorithm only converges to an approximately correct distribution, but the approximation error can be made very small.

If the sum in λ was only computed over a subset of the data, the algorithm would take less computation to run, and would be less sensitive to changes in the data.

The latter property is called *subsampling amplification* of differential privacy [WBK19]. Using the λ computed with subsampling instead of the full data λ introduces an additional error that must be corrected for to have the algorithm converge to the correct distribution.

The *central limit theorem* (CLT) states that the distribution of a sum of random variables approaches a Gaussian distribution as more random variables are summed, if some conditions on the independence and variance of the random variables are met [HJDH19]. With the CLT, it can be argued that the error from using the subsampled λ instead of the full data λ has an approximately Gaussian distribution, if the subsample is large enough [HJDH19].

The variance of the error from subsampling can be estimated by the sample variance of the individual terms in the sum in λ [HJDH19]. This allows combining the errors from subsampling and the Gaussian noise from the Gaussian mechanism to a single Gaussian noise value. The V_{corr} distribution can then be used to approximate the Barker acceptance test as above. See algorithm 2 for the DP Barker algorithm. *

Heikkilä et. al. [HJDH19] do not directly bound the sensitivity of λ as is done in DP penalty, because the sample variance also depends on input data. Instead they directly bound the Rényi divergence between $\mathcal{N}(0, \sigma^2 - \sigma_b^2)$, where σ_b^2 is the batch sample variance, for two adjacent inputs. Subsampling amplification is accounted for with an amplification theorem for Rényi DP [WBK19].

Theorem 8. *If*

$$|\ln p(x_j \mid \theta') - \ln p(x_j \mid \theta)| \leq \frac{\sqrt{|B|}}{n}$$

$$\alpha < \frac{|B|}{5}, \alpha \in \mathbb{N}$$

for all $\theta, \theta' \in \Theta$, all X and $1 \leq j \leq n$, running k iterations of DP Barker is $(\alpha, k\epsilon(\alpha))$-RDP, with

$$\epsilon(\alpha) = \frac{1}{\alpha - 1} \ln \left(1 + q^2 \binom{\alpha}{2} \min\{4(e^{\epsilon'(2)} - 1), 2e^{\epsilon'(2)}\} + 2 \sum_{j=3}^{\alpha} q^j \binom{\alpha}{j} e^{(j-1)\epsilon'(j)} \right)$$

and

$$\epsilon'(\alpha) = \frac{5}{2|B|} + \frac{1}{2(\alpha - 1)} \ln \frac{2|B|}{|B| - 5\alpha} + \frac{2\alpha}{|B| - 5\alpha}$$

where n is the number of rows in D, $|B|$ is the size of the minibatch and $q = \frac{|B|}{n}$.

Like DP penalty, DP Barker requires a bound on the log likelihood ratio for one row of data. The bound can be forced through clipping if the model does not meet it, but because of the n in the denominator of the bound, it can get very tight for large

*See [HJDH19] for the sampling procedure of V_{corr}.

values of n. As a result, clipping may be needed for almost all log likelihood ratios, which may cause the algorithm to converge to a very different distribution from the posterior.

To alleviate the tight bound on log likelihood sensitivity, DP Barker is best used with a tempered likelihood [HJDH19]. In tempering, the log likelihood is multiplied by a number $T = \frac{n_0}{n} < 1$. This increases the variance of the resulting posterior and may lower modeling error in some cases [HJDH19].

Using the tempered likelihood, the log likelihood bound becomes

$$T|\ln p(x_j \mid \theta') - \ln p(x_j \mid \theta)| \leq \frac{\sqrt{|B|}}{n}$$

which is equivalent to

$$|\ln p(x_j \mid \theta') - \ln p(x_j \mid \theta)| \leq \frac{\sqrt{|B|}}{n_0}$$

Typically $n_0 \ll n$ for large datasets, so using a tempered likelihood requires significantly less clipping than a nontempered likelihood.

Algorithm 2: DP Barker

for $1 \leq i \leq k$ **do**
- denote $\theta = \theta$
- sample $\theta' \sim q(\theta)$
- sample $B \subset \{1, \ldots, n\}$
- **for** $j \in B$ **do**
  - $r_j = \ln p(\theta' \mid x_j) - \ln p(\theta \mid x_j)$
- **end**
- $\sigma_b^2 = \mathrm{Var}\{r_j \mid j \in B\}$
- $\lambda = \frac{n}{|B|} \sum_{j \in B} r_j + \ln \frac{p(\theta')}{p(\theta)} + \ln \frac{q(\theta|\theta')}{q(\theta'|\theta)}$
- sample $s \sim \mathcal{N}(0, \sigma^2 - \sigma_b^2)$
- sample $c \sim V_{corr}^{\sigma^2}$
- $\theta_i = \begin{cases} \theta' & \text{if } \lambda + s + c > 0 \\ \theta & \text{otherwise} \end{cases}$

end

return $(\theta_1, \ldots, \theta_k)$

3.3 Comparing DP Penalty and DP Barker

4. Variations of the Penalty Algorithm

4.1 The Penalty Algorithm with Subsampling

In the DP Barker algorithm, the log likelihood ratio is computed using only a subsample of the dataset to amplify privacy. Subsampling can also be used with the penalty algorithm in the same way, if the acceptance test is corrected for the subsampling.

As with DP Barker, the error from subsampling is approximately normally distributed by the central limit theorem. The variance of the subsampling error can be estimated from the sample variance of individual terms of the sum in the log likelihood ratio. This means that the penalty method can be used to correct for the subsampling error.

The acceptance probability with subsampling is

$$\min\{1, e^{\lambda^*(\theta,\theta') - \frac{1}{2}(\sigma^2+\sigma_b^2)}\},$$

where

$$\lambda^*(\theta,\theta') = \frac{nT}{|B|}\sum_{j\in B}\ln\frac{p(x_j \mid \theta')}{p(x_j \mid \theta)} + \ln\frac{p(\theta')q(\theta \mid \theta')}{p(\theta)q(\theta' \mid \theta)},$$

and σ_b^2 is the sample variance of the log likelihood ratios in batch B. Denote

$$r_j = \ln\frac{p(x_j \mid \theta')}{p(x_j \mid \theta)},$$

$$R = \sum_{x\in B} r_j.$$

Then σ_b^2 can be estimated from the sample variance of r_j:

$$\begin{aligned}\sigma_b^2 &= \mathrm{Var}\left(\frac{nT}{|B|}\sum_{j\in B} r_j\right) = \frac{nT^2}{|B|^2}\sum_{j\in B}\mathrm{Var}(r_j) = \frac{nT^2}{|B|}\mathrm{Var}(r_j)\\ &\approx \frac{(nT)^2}{|B|^2}\sum_{j\in B}\left(r_j - \frac{R}{|B|}\right)^2 = \frac{(nT)^2}{|B|^2}\left(\sum_{j\in B} r_j^2 - \frac{R^2}{|B|}\right).\end{aligned}$$

Because σ_b^2 depends on the data, releasing λ privately is not enough, $\lambda - \frac{1}{2}\sigma_b^2$ must be released privately. This means that using subsampling requires adding additional noise to account for the sensitivity of $\frac{1}{2}\sigma_b^2$.

The sensitivity of $\lambda - \frac{1}{2}\sigma_b^2$ is

$$\Delta\lambda + \frac{1}{2}\Delta\sigma_b^2.$$

With the bound $r_j \leq L||\theta - \theta'||_2$ used in DP penalty, the bound sensitivity of λ is the same as without subsampling. The sensitivity of σ_b^2 must be bounded separately.

Lemma 1. *The sensitivity of $\frac{1}{2}\sigma_b^2$, with $r_j \leq L||\theta - \theta'||_2$, has upper bound*

$$\frac{1}{2}\Delta\sigma_b^2 \leq \left(\frac{nT}{b}\right)^2 \left|1 - \frac{1}{b}\right| L^2||\theta - \theta'||_2^2 + \frac{2(b-1)}{b}\left(\frac{nT}{b}\right)^2 L^2||\theta - \theta'||_2^2.$$

Proof. For datasets $X \sim X'$, that only differ in one element, denote the common part they have by X^*, and the differing element by $x \in X$ and $x' \in X'$

$$\begin{aligned}
\Delta\sigma_b^2 &= \sup_{D\sim D'} |\sigma_b^2(X) - \sigma_b^2(X')| \\
&= \left(\frac{nT}{b}\right)^2 \sup_{X\sim X'} \left| \sum_{x\in X} r^2(x) - \sum_{x\in X'} r^2(x) + \frac{1}{b}R^2(X') - \frac{1}{b}R^2(X)\right| \\
&= \left(\frac{nT}{b}\right)^2 \sup_{x,x',X^*} \left| r^2(x) - r^2(x') + \frac{1}{b}(R(X^*) + r(x'))^2 - \frac{1}{b}(R(X^*) + r(x))^2\right| \\
&= \left(\frac{nT}{b}\right)^2 \sup_{x,x',X^*} \Big| r^2(x) - r^2(x') + \frac{1}{b}(R^2(X^*) + 2R(X^*)r(x') + r^2(x')) \\
&\quad - \frac{1}{b}(R^2(X^*) + 2R(X^*)r(x) + r^2(x))\Big| \\
&= \left(\frac{nT}{b}\right)^2 \sup_{x,x',X^*} \left| \left(1 - \frac{1}{b}\right)(r^2(x) - r^2(x')) + \frac{2}{b}D(X^*)(r(x') - r(x))\right| \\
&\leq \left(\frac{nT}{b}\right)^2 \left|1 - \frac{1}{b}\right| \sup_{x,x'} \left|(r^2(x) - r^2(x'))\right| + \frac{2}{b}\left(\frac{nT}{b}\right)^2 \sup_{x,x',X^*} \left|R(X^*)(r(x') - r(x))\right| \\
&= \left(\frac{nT}{b}\right)^2 \left|1 - \frac{1}{b}\right| \sup_{x,x'} \left|(r^2(x) - r^2(x'))\right| + \frac{2}{b}\left(\frac{nT}{b}\right)^2 \sup_{x,x'} |r(x') - r(x)| \sup_{X^*} |R(X^*)| \\
&\leq \left(\frac{nT}{b}\right)^2 \left|1 - \frac{1}{b}\right| \sup_{x,x'} \left|(r^2(x) - r^2(x'))\right| + \frac{2}{b}\left(\frac{nT}{b}\right)^2 \sup_{x,x'} |r(x') - r(x)|(b-1) \sup_{d} |r(x)|.
\end{aligned}$$

Plugging the bound $\sup_x |r(x)| \leq L||\theta - \theta'||_2$ into the last expression proves the claim. □

Theorem 9. *Let*

$$\Delta_\lambda = \frac{2nTL}{|B|}||\theta - \theta'||_2,$$

$$\Delta_\sigma = \left(\frac{nT}{b}\right)^2 \left|1 - \frac{1}{b}\right| L^2||\theta - \theta'||_2^2, + \frac{2(b-1)}{b}\left(\frac{nT}{b}\right)^2 L^2||\theta - \theta'||_2^2.$$

$$c(\theta, \theta') = \Delta_\lambda + \Delta_\sigma,$$

$$\sigma^2(\theta, \theta') = \tau c^2(\theta, \theta').$$

Then running DP penalty with subsampling for k iterations is $(\alpha, k\epsilon(\alpha))$-RDP, with

$$\epsilon(\alpha) = \frac{1}{\alpha - 1} \ln\left(1 + q^2\binom{\alpha}{2}\min\{4(e^{\epsilon'(2)} - 1), 2e^{\epsilon'(2)}\} + 2\sum_{j=3}^{\alpha} q^j \binom{\alpha}{j} e^{(j-1)\epsilon'(j)}\right),$$

and

$$\epsilon'(\alpha) = \frac{\alpha}{2\tau},$$

where n is the number of rows in D, $|B|$ is the size of the minibatch and $q = \frac{|B|}{n}$.

Proof. By Lemma 1, $\Delta_\sigma(\theta, \theta')$ an upper bound to the sensitivity of $\frac{1}{2}\sigma_b^2$, therefore $c(\theta, \theta')$ is an upper bound to the sensitivity of $\lambda - \frac{1}{2}\sigma_b^2$.

This means that a Gaussian mechanism taking a subsample B of the data as input and uses $\sigma(\theta, \theta')$ as the noise variance is $(\alpha, \epsilon'(\alpha))$-RDP with

$$\epsilon'(\alpha) = \frac{\alpha}{2\tau}.$$

By the subsampling amplification theorem [WBK19, Theorem 9] and the composition theorem of RDP (Theorem 2), the combination of subsampling and Gaussian mechanism is $(\alpha, k\epsilon(\alpha))$-RDP with

$$\epsilon(\alpha) = \frac{1}{\alpha - 1} \ln\left(1 + q^2\binom{\alpha}{2}\min\{4(e^{\epsilon'(2)} - 1), 2e^{\epsilon'(2)}\} + 2\sum_{j=3}^{\alpha} q^j \binom{\alpha}{j} e^{(j-1)\epsilon'(j)}\right)$$

when run for k iterations for integer $\alpha \geq 2$. □

The appearance of n as a multiplier of Δ_λ and n^2 as a multiplier of Δ_σ causes the noise variance to increase with n, without a corresponding decrease in ϵ'. This makes subsampled DP penalty unsuitable for problems with large datasets, unless tempering is used. Like with DP Barker, using tempering $T = \frac{n_0}{n}$ cancels n out of the noise variance.

5. Differentially Private Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC[†]) [Nea12] is a widely used MCMC algorithm that, like Metropolis-Hastings and the penalty algorithm, was originally developed for simulations in physics and chemistry, but has since been applied to Bayesian inference.

HMC simulates the trajectory of a moving particle in a way that allows it to draw samples from a target distribution [Nea12]. With perfect simulation an acceptance test would not be needed, but exact simulation of the moving particle is typically not possible so a Metropolis-Hastings acceptance test is used for proposal generated by the simulation. Despite imperfections, the simulation can generate long jumps that stay in areas of high probability, giving HMC a higher acceptance rate that Metropolis-Hastings using a Gaussian distribution as the proposal, at the cost of higher computational requirements.

This chapter first introduces HMC in the non-DP setting in Section 5.1. In Section 5.2 the HMC is made differentially private using the acceptance test of the penalty algorithm and adding noise to the proposal phase appropriately.

5.1 MCMC with Hamiltonian Dynamics

The motion of a particle on a frictionless surface of varying height is governed by the initial position and velocity of the particle, and the height of the surface at different positions [Nea12]. The kinetic energy of the particle with momentum[‡] $p \in \mathbb{R}^2$ and mass $m \in \mathbb{R}$ is $\frac{p^T p}{2m}$. The potential energy of the particle at position $\theta \in \mathbb{R}^2$ is proportional to the height of the surface, given by a continuously differentiable function $U(\theta)$. The sum of potential and kinetic energies, the total energy of the system, is called the Hamiltonian H. Hamilton's equations give the equations of motion for a system with

[†]HMC originally stood for hybrid Monte Carlo, but the name Hamiltonian Monte Carlo has become more common.

[‡]Momentum is velocity times mass.

a given Hamiltonian:

$$\frac{d\theta}{dt} = \frac{\partial H}{\partial p}, \qquad \frac{dp}{dt} = -\frac{\partial H}{\partial \theta}.$$

Given $H(\theta, p) = U(\theta) + \frac{p^T p}{2m}$, Hamilton's equations become

$$\begin{aligned}\frac{d\theta}{dt} &= \frac{p}{m},\\ \frac{dp}{dt} &= -\nabla U(\theta).\end{aligned}$$

These equations define a function T_t taking an initial state (θ, p) of the particle to the state $T(\theta, p)$ after time t. Solving T_t analytically is usually not possible, but T_t can be approximately simulated by the leapfrog method. The leapfrog method sequentially updates an estimate (θ_i, p_i) of the state in steps L of η as follows:

$$\begin{aligned}p_{i+1/2} &= p_i - \frac{\eta}{2}\nabla U(\theta_i)\\ \theta_{i+1} &= \theta_i + \frac{\eta p_{i+1/2}}{m}\\ p_{i+1} &= p_{i+1/2} - \frac{\eta}{2}\nabla U(\theta_{i+1})\end{aligned}$$

The simulation starts with $\theta_0 = \theta$ and $p_0 = p$ and the result is given by $T_{L\eta}(\theta, p) \approx (\theta_L, p_L)$.

The above equations for two-dimensional θ and p generalize to any number of dimensions d [Nea12]. The physical analogue would be a particle moving on a d-dimensional hypersurface with "height" given by $U(\theta)$. The mass of the particle, m, can also be generalized to any positive definite matrix $M \in \mathbb{R}^{p\times p}$, which changes division by m to multiplication by M^{-1}. Specifically, the kinetic energy of the particle becomes $\frac{1}{2}p^T M^{-1} p$ and $\frac{d\theta}{dt} = M^{-1}p$. The generalization to a mass matrix does not have a physical analogue, but it can be useful for MCMC.

Hamiltonian dynamics have three important properties [Nea12] that make them useful as a proposal mechanism for MCMC. They are reversibility, conservation of the Hamiltonian and conservation of volume.

Reversibility means that the function T_t is injective, meaning that it has an inverse T_{-t} where $T_t(\theta_1, p_1) = (\theta_2, p_2)$ implies that $T_{-t}(\theta_2, p_2) = (\theta_1, p_1)$ [Nea12]. For the Hamiltonian of the particle moving on a surface, the inverse is obtained by $T_t(\theta_2, -p_2) = (\theta_1, -p_1)$.

Conservation of the Hamiltonian follows from Hamilton's equations and the chain rule of derivatives:

$$\frac{dH}{dt} = \frac{\partial H}{\partial \theta}\frac{d\theta}{dt} + \frac{\partial H}{\partial p}\frac{dp}{dt} = \frac{\partial H}{\partial \theta}\frac{\partial H}{\partial p} - \frac{\partial H}{\partial p}\frac{\partial H}{\partial \theta} = 0.$$

Conservation of volume in the (θ, p)-space follows from the fact that the absolute value of the Jacobian determinant of T_t is 1 and the injectivity of T_t. A somewhat lengthier proof for the Jacobian determinant of T_t is given by Neal [Nea12].

To see why having $|\det J_f| = 1$, where J_f is the Jacobian matrix of an injective continuously differentiable function f, implies f preserving volume, recall that the volume of a set A is given by

$$\int_A 1 dx.$$

The volume of the image of A under f, $f(A)$, is given by

$$\int_{f(A)} 1 dx = \int_A |\det J_f(x)| dx = \int_A 1 dx.$$

where the first equality follows from the change of variables formula.

The leapfrog method does not conserve the Hamiltonian exactly, but it preserves volumes and is reversible exactly [Nea12]. These properties of the leapfrog method are important for its use as a proposal mechanism for MCMC.

The HMC algorithm samples from the distribution of [Nea12]

$$P(\theta, p) \propto \exp(-H(\theta, p)).$$

With $H(\theta, p) = U(\theta) + \frac{1}{2} p^T M^{-1} p$,

$$P(\theta, p) \propto \exp(-U(\theta)) \exp\left(\frac{1}{2} p^T M^{-1} p\right)$$

This means that the marginal distributions of θ and p are independent, the marginal distribution of p is a d-dimensional Gaussian distribution with mean 0 and covariance M, and the marginal distribution of θ can have any continuously differentiable log likelihood $-U$. Because of this, θ is used to represent the variables of interest, while p is a set of auxiliary variables used for the proposals.

Generating a (θ, p)-sample is done in two stages [Nea12], which are technically two separate Metropolis-Hastings proposals and acceptance tests. In the first stage, p is proposed from the Gaussian distribution with mean 0 and covariance M. Because the proposal matches the marginal distribution of p, it is always accepted.

In the second stage, Hamiltonian dynamics for the particle on a surface are simulated with the leapfrog method, starting from the current value of θ and the proposed value of p. The end result is a proposal for both θ and p. Because the leapfrog simulation preserves volumes and is reversible by negating p, the acceptance probability of the proposal is simply

$$\min\{1, \exp(H(\theta, p) - H(\theta', p'))\}.$$

If the simulation conserved H exactly, the acceptance probability would always be 1. Because this is usually not possible, there is always some probability of rejecting, which

depends on the length of the jump taken by the leapfrog method, η, and the number of leapfrog steps, L. It might be expected that the errors from the leapfrog steps would accumulate during the simulation, but in practice the errors in different steps tend to cancel each other [Nea12], meaning that the acceptance probability mainly depends on η.

5.2 Differential Privacy with HMC

HMC only uses the model log likelihood, and thus the data, through U. U is used in two ways. Firstly, its value is used in the acceptance test, and secondly, its gradients are used in to obtain proposals. Adding appropriate amounts of noise to both accesses would make HMC private, but the addition of noise may break some of the properties required for the algorithm's correctness.

The addition of noise to the log likelihood ratios can be corrected using the penalty acceptance test, i.e. changing the acceptance probability to

$$\min\left\{\exp\left(H(\theta,p) - H(\theta',p') - \frac{1}{2}\sigma_l^2\right)\right\},$$

where σ_l^2 is the variance of the noise added to the log likelihood ratios. This is because with $U(\theta) = -\ln p(X \mid \theta)$, the difference of the Hamiltonians in the acceptance probability is

$$H(\theta,p) - H(\theta',p') = \ln p(X \mid \theta') - \ln p(X \mid \theta) + \frac{1}{2}(p^T M^{-1} p - p'^T M^{-1} p').$$

As with the penalty algorithm, the sensitivity of the log likelihood ratios must be bounded. If a bound does not exist, clipping must be used, nullifies the asymptotic converge guarantee of the algorithm. However, in Section 7.2 it is shown that clipping low numbers of gradients does not affect convergence in practice.

Releasing the gradients privately requires a trade-off. The leapfrog method remains reversible with noisy and potentially clipped gradients, as is shows in the following, but they no longer simulate the correct Hamiltonian dynamics [CFG14]. This does not affect asymptotic convergence, but it can potentially lower acceptance rates. The dynamics can be corrected by adding a friction term to the equations of motion of the simulation [CFG14], but this removes volume preservation of the simulation, which means that the acceptance probability must be corrected.

The leapfrog update equations with noisy and clipped gradients become

$$\begin{aligned} p_{i+1/2} &= p_i - \frac{\eta}{2}(\text{clip}(\nabla U(\theta_i)) + \mathcal{N}(0,\sigma_g^2)) \\ \theta_{i+1} &= \theta_i + \eta M^{-1} p_{i+1/2} \\ p_{i+1} &= p_{i+1/2} - \frac{\eta}{2}(\text{clip}(\nabla U(\theta_{i+1})) + \mathcal{N}(0,\sigma_g^2)). \end{aligned}$$

The second equation is unchanged, and remains both volume preserving and reversible by negating p. Both the first and third equations apply the function $l_p \colon \mathbb{R}^{2d} \to \mathbb{R}^{2d}$,

$$l_p(\theta, p) = \left(\theta, p - \frac{\eta}{2}(\text{clip}(\nabla U(\theta)) + \mathcal{N}(0, \sigma_{g^2}))\right).$$

to θ and p.

The Jacobian of l_p is

$$J_{l_p}(\theta, p) = \begin{pmatrix} I_{d\times d} & 0_{d\times d} \\ C(\theta, p) & I_{d\times d} \end{pmatrix},$$

where $I_{d\times d}$ and $0_{d\times d}$ are the d-by-d identity and zero matrices, respectively, and $C(\theta, p) = J_{\text{clip}(\nabla U(\theta))}(\theta, p)$. Because $J_{l_p}(\theta, p)$ is triangular, $\det J_{l_p}(\theta, p) = 1$ for all θ and p, which means that l_p preserves volume. As the step updating θ also preserves volume, the entire leapfrog simulation preserves volume.

Showing that the leapfrog simulation is reversible by negating p requires more thought, as the simulation is no longer deterministic. Recall that the MH acceptance test uses the acceptance probability

$$\min\left\{1, \frac{\pi(\theta')}{\pi(\theta)}\frac{q(\theta \mid \theta')}{q(\theta' \mid \theta)}\right\},$$

where π is the target density and q is the proposal density. For HMC, q must be understood as a Dirac δ function, as the standard leapfrog is deterministic. The reversibility of the leapfrog by negating p means that $q(\theta', p' \mid \theta, p) = q(\theta, -p \mid \theta', -p')$. If p were negated after the leapfrog simulation, one would have another proposal distribution q', with $q'(\theta', p' \mid \theta, p) = q'(\theta, p \mid \theta', p')$. Running HMC with proposal q' instead of q does not change the output of the algorithm, as p is not stored and the kinetic energy term in the Hamiltonian is $\frac{1}{2}p^T M^{-1} p$, so $H(\theta, p) = H(\theta, -p)$. This means that the proposal ratio in the MH acceptance probability is one for HMC.

With the noisy gradient, the leapfrog proposal is no longer deterministic, so the reversibility by negating p must be proven separately.

The proof requires defining some new terminology. Let $l \colon \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ be a function giving a potentially random proposal for given (θ, p). Because l is potentially random, it has a (generalized) density $q_l(\theta', p' \mid \theta, p)$, where

$$P(l(\theta, p) \in A) = \int_A d(\theta', p') q_l(\theta', p' \mid \theta, p).$$

Now reversibility by negating p can be defined:

Definition 9. *A proposal function l is reversible by negating p if*

$$q_l(\theta', p' \mid \theta, p) = q_l(\theta, -p \mid \theta', -p')$$

This can be simplified by defining $\bar{l}$ as the the proposal function that has density $q_{\bar{l}}(\theta', p' \mid \theta, p) = q_l(\theta, p \mid \theta', p')$, and defining l_- as $l_-(\theta, p) = (\theta, -p)$. Definition 9 can then be written as $\bar{l} = l_- \circ l \circ l_-$.

Lemma 2. *Let l_1 and l_2 be proposal functions in $\mathbb{R}^{2d}$. Then $\overline{l_2 \circ l_1} = \bar{l_1} \circ \bar{l_2}$. For a sequence $l_1, \ldots, l_k$ of proposal functions, $\overline{l_k \circ \cdots \circ l_1} = \bar{l_1} \circ \cdots \circ \bar{l_k}$.*

Proof.

$$\begin{aligned} q_{\overline{l_2 \circ l_1}}(\theta_3, p_3 \mid \theta_1, p_1) &= q_{l_2 \circ l_1}(\theta_1, p_1 \mid \theta_3, p_3) \\ &= \int_{\mathbb{R}^{2d}} d(\theta_2, p_2) q_{l_1}(\theta_2, p_2 \mid \theta_3, p_3) q_{l_2}(\theta_1, p_1 \mid \theta_2, p_2) \\ &= \int_{\mathbb{R}^{2d}} d(\theta_2, p_2) q_{\bar{l_1}}(\theta_3, p_3 \mid \theta_2, p_2) q_{\bar{l_2}}(\theta_2, p_2 \mid \theta_1, p_1) \\ &= q_{\bar{l_1} \circ \bar{l_2}}(\theta_3, p_3 \mid \theta_1, p_1) \end{aligned}$$

which means that $\overline{l_2 \circ l_1} = \bar{l_1} \circ \bar{l_2}$. The claim for sequences follows by induction. □

Theorem 10. *The leapfrog simulation with noisy and clipped gradients is reversible by negating p.*

Proof. The leapfrog simulation is a composition of proposal functions l_p and l_θ where

$$l_p(\theta, p) = \left(\theta, p - \frac{\eta}{2}(\text{clip}(\nabla U(\theta)) + \mathcal{N}(0, \sigma_g^2))\right)$$

and

$$l_\theta(\theta, p) = (\theta + \eta M^{-1} p, p).$$

Specifically, the simulation is

$$(l_p \circ l_\theta \circ l_p) \circ \cdots \circ (l_p \circ l_\theta \circ l_p)$$

with L compositions of $l_p \circ l_\theta \circ l_p$. Denoting $g(\theta) = \text{clip}(\nabla U(\theta))$,

$$\begin{aligned} q_{l_p}(\theta', p' \mid \theta, p) &= \delta_{\theta'}(\theta) \cdot \mathcal{N}\left(p' \mid p - \frac{\eta}{2} g(\theta), \frac{\eta^2}{4}\sigma_g^2\right) \\ &= \delta_\theta(\theta') \cdot \mathcal{N}\left(-p \mid -p' - \frac{\eta}{2} g(\theta), \frac{\eta^2}{4}\sigma_g^2\right) \\ &= q_{l_p}(\theta, -p \mid \theta', -p') \end{aligned}$$

and

$$\begin{aligned} q_{l_\theta}(\theta, -p \mid \theta', -p') &= \delta_\theta(\theta' - \eta M^{-1} p') \cdot \delta_{-p}(-p') \\ &= \delta_{\theta'}(\theta + \eta M^{-1} p) \cdot \delta_{p'}(p) \\ &= q_{l_\theta}(\theta', p' \mid \theta, p), \end{aligned}$$

which means that both l_p and l_θ are reversible by negating p. Then the reverse of the leapfrog composition can be written as

$$\begin{aligned}\overline{(l_p \circ l_\theta \circ l_p) \circ \cdots \circ (l_p \circ l_\theta \circ l_p)} &= (\bar{l_p} \circ \bar{l_\theta} \circ \bar{l_p}) \circ \cdots \circ (\bar{l_p} \circ \bar{l_\theta} \circ \bar{l_p}) \\ &= l_- \circ (l_p \circ l_\theta \circ l_p) \circ \cdots \circ (l_p \circ l_\theta \circ l_p) \circ l_-,\end{aligned}$$

where the first equality uses Lemma 2 and the fact that the sequence of compositions is symmetric. The second equality uses the fact that l_p and l_θ are reversible by negating p. This means that the leapfrog simulation with noisy and clipped gradients is reversible by negating p. □

If the friction term that corrects the dynamics is used, the equations of motion become [CFG14]

$$\begin{aligned}\frac{d\theta}{dt} &= M^{-1}p \\ \frac{dp}{dt} &= -\nabla U(\theta) - \frac{1}{2}\sigma_g^2 M^{-1}p + \mathcal{N}(0, \sigma_g^2)\end{aligned}$$

The equation for θ is unchanged, but the equation for p has the additional term $-\frac{1}{2}\sigma_g^2 M^{-1}p$. For the leapfrog updates, l_θ does not change, and l_p changes to

$$l_p(\theta, p) = \left(\theta, p - \frac{\eta}{2}(\text{clip}(\nabla U(\theta)) + \frac{1}{2}\sigma_g^2 M^{-1}p + \mathcal{N}(0, \sigma_g^2))\right).$$

l_p no longer preserves volumes, as the Jacobian of l_p is

$$J_{l_p}(\theta, p) = \begin{bmatrix} I_{d\times d} & 0_{d\times d} \\ C(\theta, p) & I_{d\times d} - \frac{\eta}{4}\sigma_g^2 M^{-1} \end{bmatrix}$$

which means that generally $|\det J_{l_p}(\theta, p)| \neq 1$.

DP HMC is shown in Algorithm 3.

Algorithm 3: DP HMC

$G_{1,0} = \text{grad}(\theta_0, b_g)$
for $1 \leq i \leq k$ **do**
  $p_0 = \mathcal{N}_d(0, M)$
  $\theta_{i,0} = \theta_{i-1}$
  for $1 \leq j \leq L$ **do**
    $p_{j-0.5} = p_{j-1} + \frac{1}{2}\eta\left(G_{i,j-1} - \frac{1}{2}\sigma_g^2 M^{-1} p_{j-1}\right)$
    $\theta_{i,j} = \theta_{i,j-1} + M^{-1} p_{j-0.5}$
    $G_{i,j} = \text{grad}(\theta_{i,j}, b_g)$
    $p_j = p_{j-0.5} + \frac{1}{2}\eta\left(G_{i,j} - \frac{1}{2}\sigma_g^2 M^{-1} p_{j-0.5}\right)$
  end
  $r_l = \ln \frac{p(\theta_{i,L}|x_l)}{p(\theta_{i-1}|x_l)}$
  $R = \sum_{l=1}^{n} \text{clip}_{b_l||\theta_{i-1}-\theta_{i,L}||_2}(r_l)$
  $s_{var} = (2\sigma_l b_l ||\theta_{i-1} - \theta_{i,L}||_2)^2$
  $\Delta H = R + \frac{1}{2} p_0^T M^{-1} p_0 - \frac{1}{2} p_L^T M^{-1} p_L + \ln \frac{p(\theta_{i,L})}{p(\theta_{i-1})} + \mathcal{N}(0, s_{var})$
  $u = \text{Unif}(0, 1)$
  if $u < \Delta H - \frac{1}{2} s_{var}$ **then**
    $\theta_i = \theta_{i,L}$
    $G_{i+1,0} = G_{i,L}$
  end
  else
    $\theta_i = \theta_{i-1}$
    $G_{i+1,0} = G_{i,0}$
  end
end

where

$$\text{grad}(\theta, b) = \sum_{i=1}^{n} \text{clip}_b(\nabla \ln p(\theta \mid x_i)) + \nabla \ln p(\theta)$$

and $\text{clip}_b(\theta)$ clips the euclidean norm of θ to be less than or equal to b.

Each iteration of the outer for-loop computes the gradient L times and the log likelihood ratio once. The gradient is also computed once before the outer for-loop. Releasing a single gradient has zCDP privacy cost of

$$\rho_g = \frac{4b_g^2}{2\sigma_g^2} = \frac{1}{2\tau_g^2 n},$$

where $\sigma_g = 2\tau_g \sqrt{n} b_g$. Releasing the log likelihood ratio of θ and θ' has privacy cost

$$\rho_l = \frac{4b_l^2 ||\theta - \theta'||_2^2}{2\sigma_l(\theta, \theta')^2} = \frac{1}{2\tau_l^2 n},$$

where $\sigma_l(\theta, \theta') = 2\tau_l \sqrt{n} b_l ||\theta - \theta'||_2$.

The total zCDP budget with given (ϵ, δ)-bound is

$$\rho = \left(\sqrt{\epsilon - \ln \delta} - \sqrt{-\ln \delta}\right)^2.$$

This means that the number of iteration that the algorithm is allowed to run for is

$$k = \left\lfloor \frac{\rho - \rho_g}{\rho_l + L\rho_g} \right\rfloor$$

zCDP based privacy accounting is loose, to the above bound could be improved by using the tight bound for the Gaussian mechanism. Because DP HMC has both different sensitivities and different noise variances between the log likelihood ratios and gradients, Theorem 4 is not directly applicable. However, the bound of Theorem 4 is extendable to differing variances between composition.

Sommer et. al. [SMM19] prove the ADP bound of Theorem 4 with the *privacy loss distribution* (PLD) of the Gaussian mechanism. First, they show that the PLD of the Gaussian mechanism with sensitivity Δ and variance σ^2 is a Gaussian distribution $\mathcal{N}(\mu, 2\mu)$, where $\mu = \frac{\Delta^2}{2\sigma^2}$ [SMM19, Lemma 11]. Next, they show that the ADP bounds $(\epsilon, \delta(\epsilon))$ for a Gaussian PLD $\mathcal{N}(\mu, 2\mu)$ are given by [SMM19, Lemma 12]

$$\delta(\epsilon) = \frac{1}{2}\left(\operatorname{erfc}\left(\frac{\epsilon - \mu}{2\sqrt{\mu}}\right) - e^{\epsilon} \operatorname{erfc}\left(\frac{\epsilon + \mu}{2\sqrt{\mu}}\right)\right).$$

Finally, they show that the PLD of an adaptive composition is the convolution of the PLDs of the individual parts of the composition [SMM19, Theorem 1]. As convolution of distributions corresponds to summing random variables, the PLD of a convolution of Gaussian mechanisms with PLDs $\mathcal{N}(\mu_i, 2\mu_i)$ is simply $\mathcal{N}(\sum \mu_i, 2\sum \mu_i)$.

Because the sensitivity of the log likelihood ratio is $2b_l||\theta - \theta'||_2$ and the sensitivity of the gradient is $2b_g$, both the gradient and log likelihood ratio are divided by their sensitivities before adding noise, which normalises both to have sensitivity one. Adding noise with variance $\sigma_l^2(\theta, \theta')$ and σ_g^2 to the log likelihood ratio and gradient respectively is equivalent to adding noise with variance $\sigma_l'^2 = \tau_l^2 n$ and $\sigma_g'^2 = \tau_g^2 n$. to the normalised log likelihood ratio and gradient.

DP HMC releases the log likelihood ratio k times and the gradient $kL + 1$ times. This means that the PLD of DP HMC is a Gaussian distribution $\mathcal{N}(\mu, 2\mu)$ with

$$\mu = \frac{k}{2\sigma_l'^2} + \frac{kL + 1}{2\sigma_g'^2}.$$

6. Limitations and Alternatives of DP MCMC

6.1 Dataset Size

6.2 Alternatives to DP MCMC

6.2.1 DP Variational Inference

6.2.2 Releasing Summary Statistics Privately

6.2.3 MCMC with synthetic data

7. Experiments

7.1 Maximum Mean Discrepancy

The convergence of non-DP MCMC algorithms is typically assessed using $\hat{R}$ [GCS+14], which measures how well multiple chains started from different points have mixed together. The utility of a sample produced by an MCMC algorithm can be evaluated using effective sample size (ESS) [GCS+14], which is an estimate of the size of an uncorrelated sample of the posterior with the same estimation utility as the MCMC sample.

Both $\hat{R}$ and ESS require that the MCMC algorithm asymptotically targets the true posterior, as they cannot detect an algorithm that has converged to the wrong distribution. Because some of the DP MCMC algorithms use approximations that may cause the algorithms to not converge to the true posterior, such as clipping the log likelihood ratios, $\hat{R}$ and ESS are not suitable for assessing the performance of the algorithms.

Because the DP MCMC algorithms may not converge to the correct distribution, their performance should be evaluated with a metric that measures how close to the true distribution they are. A very general such metric is maximum mean discrepancy (MMD) [GBR+12]. MMD between distributions p and q is defined as

$$\mathrm{MMD}(p, q) = \sup_{f \in \mathcal{F}} (E_{x \sim p} f(x) - E_{y \sim q} f(y))$$

where $\mathcal{F}$ is some class of functions. By choosing a suitable $\mathcal{F}$, $\mathrm{MMD}(p, q)$ can be estimated from a sample from p and q. The suitable classes $\mathcal{F}$ can be characterised by a kernel function $k \colon P \times Q \to \mathbb{R}$, where P and Q are the supports of p and q, respectively. The Gaussian radial basis function (RBF) kernel

$$k(x, y) = \exp\left(\frac{||x - y||_2^2}{2\sigma^2}\right)$$

is particularly well suited, as it has the property that $\mathrm{MMD}(p, q) = 0$ if and only if $p = q$. After choosing a kernel, $\mathrm{MMD}(p, q)$ may be estimated from finite samples

of p and q. To evaluate MCMC algorithms, one of the samples is the output of the algorithm to be evaluated, and the other is a sample from the true posterior.

The choice of the σ parameter of k affects the way MMD evaluates different kinds of differences in p and q. For some preliminary experiments in this thesis, σ was chosen to be 1, which penalised error in the mean much more than errors in higher moments in the experiments. The σ used for the final experiments chosen by picking 50 subsamples from both samples with replacement and setting σ to be the median between distances of points of the subsamples. This is following the procedure of Gretton et. al. [GBR+12], with the addition of the subsampling step to handle samples of different sizes.

7.2 The Effects of Clipping

The first experiment evaluates the effect of clipping log likelihood ratios. Both HMC and random walk Metropolis-Hastings (RWMH)* algorithms were run on 2 and 10 dimensional banana models. RWMH did not converge in reasonable time in 10 dimensions so it was excluded from these results. DP was not used so that error from the extra noise would not affect the results.

The banana model used hyperparameter values $a = 20$, $b = m = 0$, $\sigma_1^2 = 20$, $\sigma_2^2 = 2.5$, $\sigma_3^2 = \cdots = \sigma_{10}^2 = 1$, $\sigma_0^2 = 1000$ and $n = 100000$. In both experiments, 500 samples from HMC and 3000 samples from RWMH were taken and the latter half† of them were compared to 2000 samples from the true posterior. The reference posterior sample was also compared to other samples from the posterior to obtain a baseline

Figure 7.1 shows the results of the clipping experiment. The top left and bottom left panels show MMD as a function of the clip bound and the fraction of log likelihoods that was actually clipped for both HMC and RWMH in the 2-dimensional model. The effect of clipping on MMD is nonexistent for all but the lowest clip bounds. The top and bottom right panels show results for the 10-dimensional model. This time there are chains that did not converge correctly with most clip bounds, but the chains with the higher bounds converged. Based on these results, if the clip fraction is less than 10%, clipping is likely undetectable without a large sample.

*Metropolis-Hastings using the Gaussian distribution as the proposal distributions.

†As the start of an MCMC chain depends heavily on the starting point, samples at the start should be discarded as they are not representative of the target distribution.

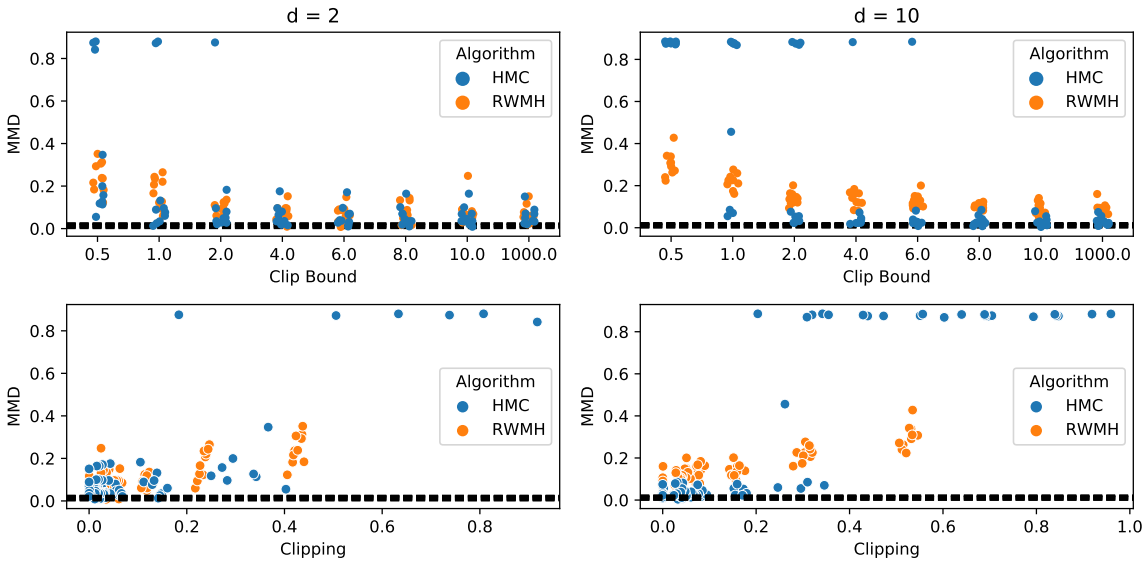


Figure 7.1: The effect of log likelihood ratio clipping on the posterior of the banana model for random walk Metropolis-Hastings and HMC. The top row shows posterior MMD as a function of the clip bound, and the bottom row shows MMD as a function of the fraction of log likelihoods that were clipped. The left columns used a 2-dimensional posterior while the right columns had a 10-dimensional posterior. The black lines show the MMDs ten different samples of the true posterior compared to the reference sample. All of the lines are close the each other and appear as a single line.

7.3 Illustrating HMC

7.4 Banana Distribution

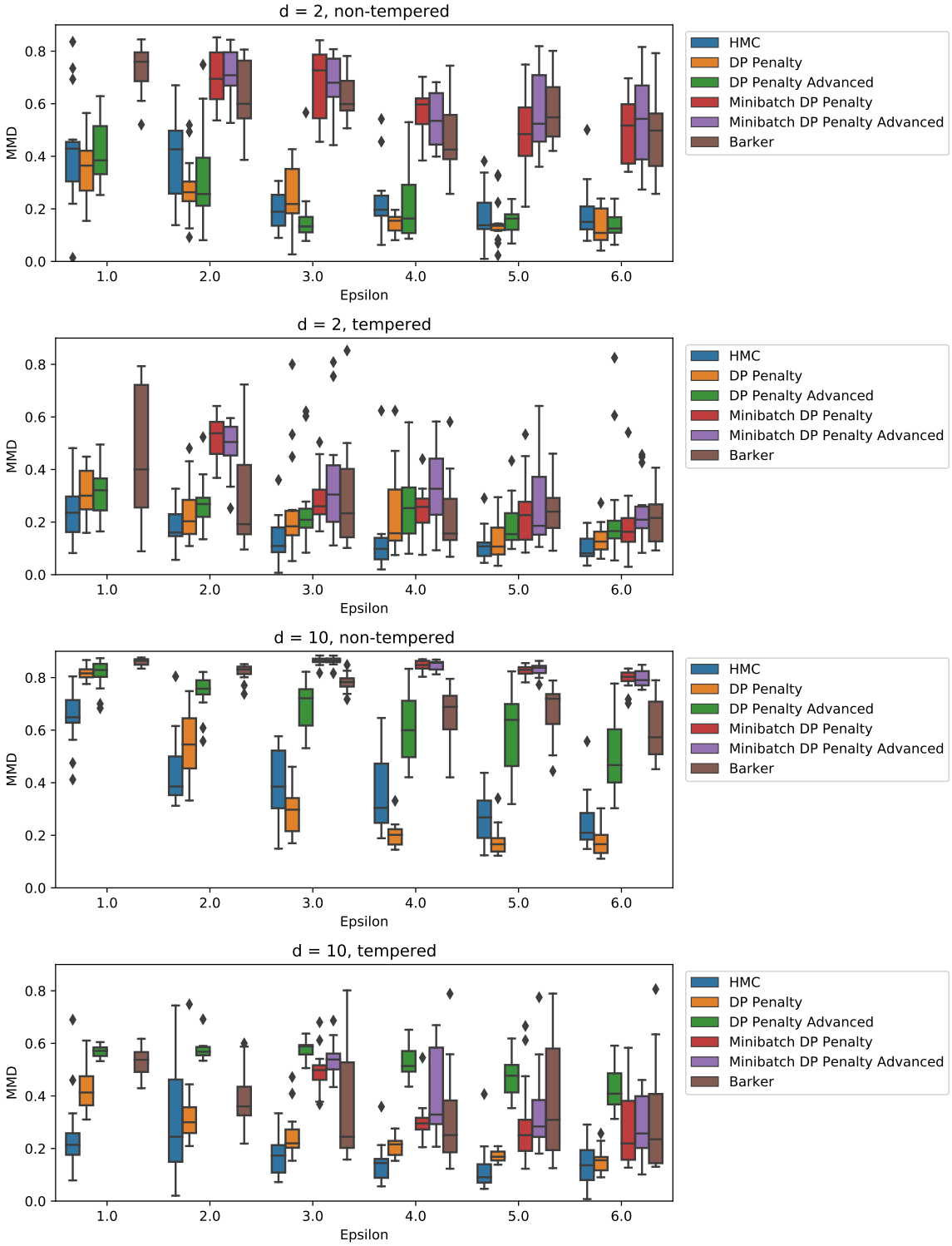
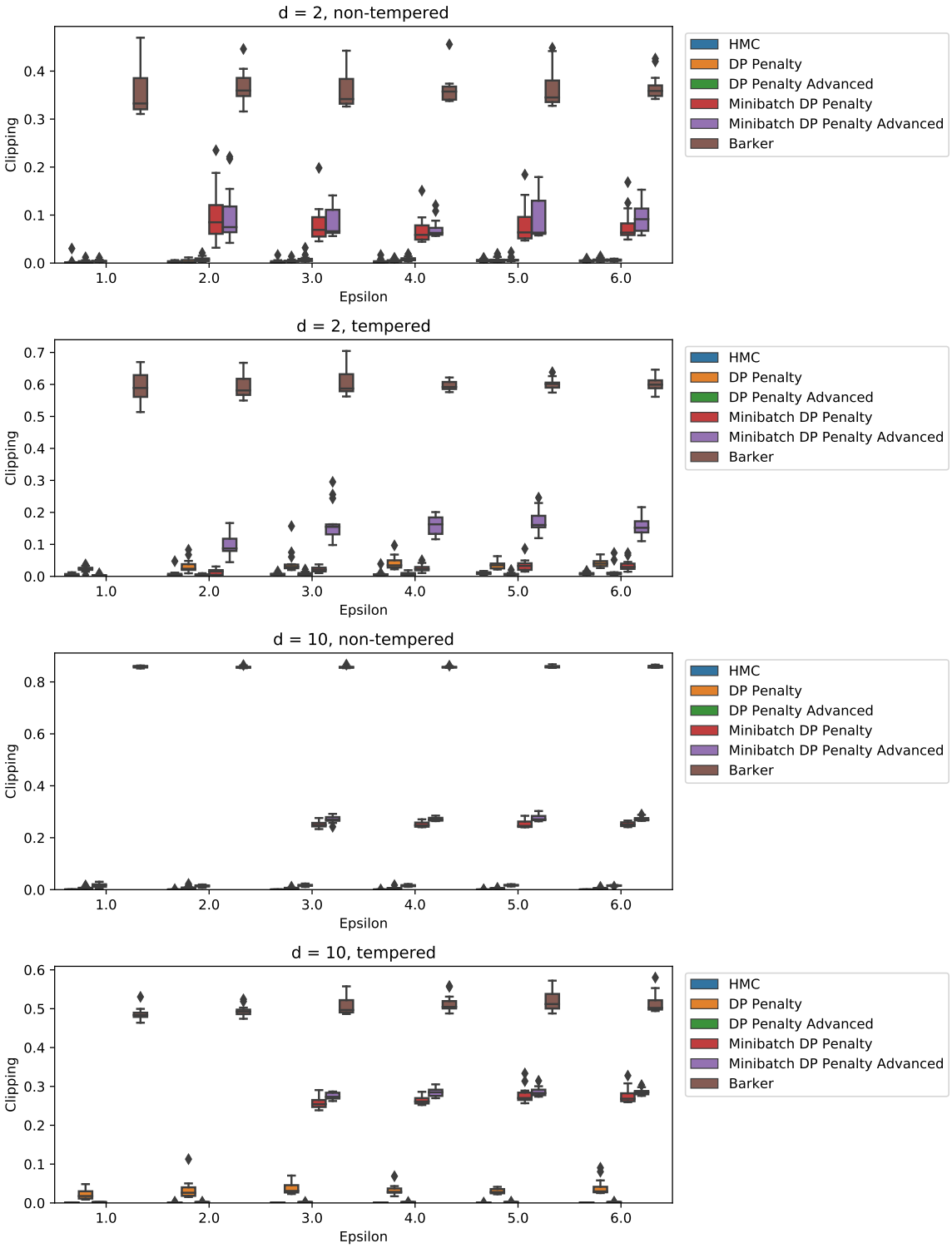
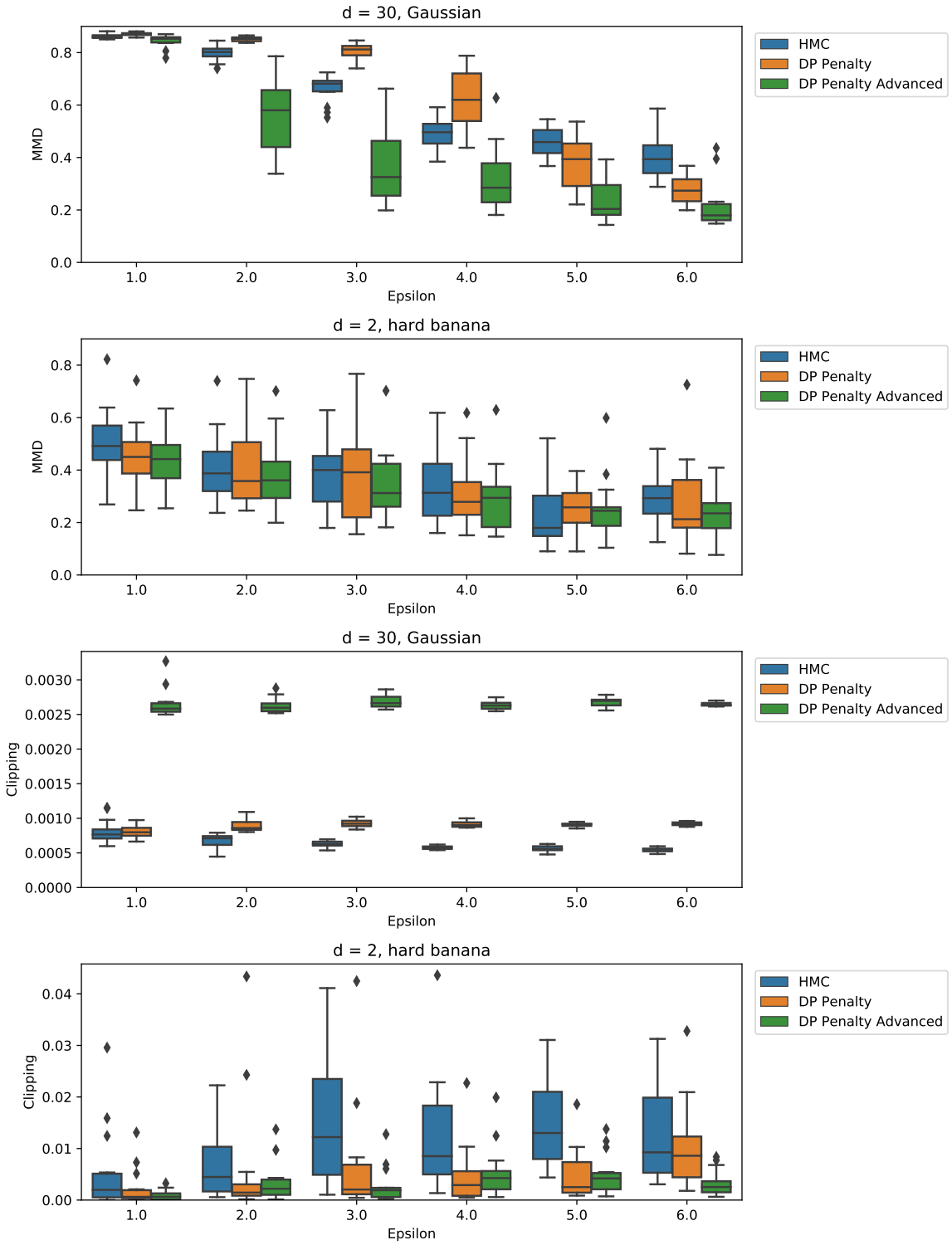


Figure 7.2: MMD as a function of ϵ for the different MCMC algorithms, with both tempering and a low number of dimensions (2) and a high number of dimensions (10).


d = 2, non-tempered
Clipping
Epsilon
d = 2, tempered
Clipping
Epsilon
d = 10, non-tempered
Clipping
Epsilon
d = 10, tempered
Clipping
Epsilon
HMC
DP Penalty
DP Penalty Advanced
Minibatch DP Penalty
Minibatch DP Penalty Advanced
Barker


d = 30, Gaussian
MMD
Epsilon
HMC
DP Penalty
DP Penalty Advanced
d = 2, hard banana
MMD
Epsilon
HMC
DP Penalty
DP Penalty Advanced
d = 30, Gaussian
Clipping
Epsilon
HMC
DP Penalty
DP Penalty Advanced
d = 2, hard banana
Clipping
Epsilon
HMC
DP Penalty
DP Penalty Advanced

8. Conclusions

Bibliography

[Bar65] Av A Barker. Monte carlo calculations of the radial distribution functions for a proton-electron plasma. *Australian Journal of Physics*, 18(2):119–134, 1965.

[BS16] Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part I*, pages 635–658, 2016.

[CD99] DM Ceperley and Mark Dewing. The penalty method for random walks with uncertain energies. *The Journal of chemical physics*, 110(20):9812–9820, 1999.

[CFG14] Tianqi Chen, Emily B. Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1683–1691. JMLR.org, 2014.

[DR14] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[GBR+12] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. A kernel two-sample test. *J. Mach. Learn. Res.*, 13:723–773, 2012.

[GCS+14] Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. Chapman & Hall/CRC texts in statistical science series. CRC Press, Boca Raton, third edition, 2014.

[HJDH19] Mikko A. Heikkilä, Joonas Jälkö, Onur Dikmen, and Antti Honkela. Differentially private markov chain monte carlo. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information*

Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada, pages 4115–4125, 2019.

[Mir17] Ilya Mironov. Rényi differential privacy. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 263–275, 2017.

[Nea12] Radford M. Neal. Mcmc using hamiltonian dynamics, 2012.

[SMM19] David M. Sommer, Sebastian Meiser, and Esfandiar Mohammadi. Privacy loss classes: The central limit theorem in differential privacy. *PoPETs*, 2019(2):245–269, 2019.

[TPK14] Minh-Ngoc Tran, Michael K. Pitt, and Robert Kohn. Adaptive metropolis-hastings sampling using reversible dependent mixture proposals. *Statistics and Computing*, 26(1-2):361–381, 2014.

[WBK19] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled renyi differential privacy and analytical moments accountant. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, pages 1226–1235, 2019.

[YE19] Sinan Yildirim and Beyza Ermis. Exact MCMC with differentially private moves - revisiting the penalty algorithm in a data privacy framework. *Statistics and Computing*, 29(5):947–963, 2019.