Privacy-Aware Rejection Sampling

Jordan Awan
Purdue University
jawan@purdue.edu

Vinayak Rao
Purdue University
varao@purdue.edu

Abstract

Differential privacy (DP) offers strong protection against adversaries with arbitrary side-information and computational power. However, many implementations of DP mechanisms leave themselves vulnerable to side channel attacks, such as timing attacks. As many privacy mechanisms, such as the exponential mechanism, do not lend themselves to easy implementations, when sampling methods such as MCMC or rejection sampling are used, the runtime can leak privacy. In this work, we quantify the privacy cost due to the runtime of a rejection sampler in terms of (ϵ, δ)-DP. We also propose three modifications to the rejection sampling algorithm, to protect against timing attacks by making the runtime independent of the data. We also use our techniques to develop an adaptive rejection sampler for log-Holder densities, which also has data-independent runtime.

CCS Concepts: • **Security and privacy → Privacy protections**; **Usability in security and privacy**; Social aspects of security and privacy.

Keywords: differential privacy, timing attack, side-channel

ACM Reference Format:
Jordan Awan and Vinayak Rao. 2021. Privacy-Aware Rejection Sampling. In *Privacy Preserving Machine Learning Workshop '21: ACM CSS 2021, November 19, 2021, Coex, Seoul, South Korea.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

As more data is collected, analyzed, and published by researchers, companies, and government agencies, concerns about the privacy of the participating individuals have become more prominent [19]. While there have been many methods of statistical disclosure control to combat this problem [15], differential privacy (DP) [10] has arisen as the state-of-the-art framework for privacy protection. Differential privacy is based on a notion of plausible deniability, and requires the introduction of additional noise, beyond sampling, into the analysis procedure. Given the output of a DP mechanism, an adversary cannot determine with high probability whether any particular individual participated in the dataset [31].

Because of the formal nature of DP, implementations of the mechanisms must be very careful to prevent unintentional privacy leaks through side-channels. Side-channel attacks have been a long-standing problem in computer systems, for example the execution time, power consumption, or memory usage of the system, are a few examples of side-channels [16, 25]. With differential privacy, the system can be made black-box to remove some of these side channels, but may still be susceptible to timing attacks. PINQ [22] and Airavat [27] were two of the earliest DP implementations, but were shown by Haeberlen et al. [13] to be vulnerable to timing attacks. FUZZ [13] and GUPT [24] avoid timing attacks by working with simple queries for which the worst-case computational time can be determined. This solution works for simple DP tasks, however, there are complex DP mechanisms for which it is nontrivial to design implementations with data-independent runtime.

A common and powerful DP mechanism is the exponential mechanism [21] which results in an unnormalized density of the form $\exp(g(x))$ that must be sampled from. The exponential mechanism has been widely used to tackle problems such as principal component analysis [4, 8, 17], K-means clustering, [11], convex optimization [6, 7], robust regression [3], linear and quantile regression [26], synthetic data [28], and Bayesian data analysis [9, 23, 30, 32].

A challenge however is that for many functions $g(x)$ encountered in practice, sampling from $\exp(g(x))$ is challenging. In statistics and machine learning, there have been many computational techniques proposed to produce either exact or approximate samples from such distributions including Markov chain Monte Carlo (MCMC), rejection sampling, approximate Bayesian computing, etc. However, there are two sources of privacy leaks when using these computational sampling methods: 1) when using approximate samplers, the resulting sample does not exactly follow the target distribution, with the error in the approximation resulting in an increased privacy risk, 2) with either an approximate or exact sampler, if the run time of the algorithm depends on the database, then this side-channel leaks privacy [13].

We consider the runtime of the algorithm as an additional output of the mechanism, and require that both the official output and the runtime jointly satisfy differential privacy.

PPML '21, November 19, 2021, Coex, Seoul, South Korea

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Contributions We quantify the privacy risk of rejection sampling as well as adaptive rejection sampling, before considering any privacy-preserving modifications. As a rejection sampler results in samples with distribution equal to the target, the only privacy concern is the runtime, which varies for different databases. We characterize the privacy risk of a simple rejection sampler in terms of (ϵ, δ)-DP. We also show that a simple rejection sampler does not satisfy ϵ-DP for any finite ϵ unless the acceptance rate is constant across databases. Similarly, an adaptive rejection sampler does not satisfy ϵ-DP unless a stringent condition holds for the acceptance probabilities across databases.

Given the increased privacy risk due to the runtime, we propose several variations on the rejection sampler, which make the runtime independent of the database: 1) choose the number of iterations to run the sampler ahead of time, 2) introduce an additive wait-time based on a worst-case dataset, 3) use squeeze functions to add an implicit wait-time. The third approach also leads to an adaptive rejection sampler which can be applied to any log-Holder density. Finally, we give examples of the exponential mechanism which satisfy the assumptions of our methods.

1.1 Differential privacy

Given a metric space $(\mathcal{D}, d)$, which represents the set of possible databases, a set of probability measures $\{M_D \mid D \in \mathcal{D}\}$ on a common space $\mathcal{Y}$ is called a *privacy mechanism*. The space $\mathcal{D}$ represents the space of possible databases. When implementing a privacy mechanism, we release one sample from M_D, which satisfies some form of privacy.

Definition 1.1 ((ϵ, δ)-DP:10)**.** Given a metric space $(\mathcal{D}, d)$, $\epsilon > 0$ and $\delta \geq 0$, a privacy mechanism $\{M_D\}$ on the space $\mathcal{Y}$ satisfies (ϵ, δ)-differential privacy if for all measurable sets $B \in \mathcal{Y}$ and all $d(D, D') \leq 1$, $M_D(B) \leq \exp(\epsilon) M_{D'}(B) + \delta$.

The values ϵ and δ are called the privacy parameters, which capture the privacy risk for the given mechanism. Smaller values of ϵ and δ give stronger privacy guarantees. We call $(\epsilon, 0)$-DP "pure differential privacy," and write ϵ-DP.

2 Privacy risk of rejection sampling

In this section we characterize the privacy cost of a rejection sampler. Recall that a rejection sampler results in an exact sample from the target distribution. Thus, the only increased privacy risk from using this algorithm is due to the runtime.

Assumption 2.1. For a rejection sampler, along with the published accepted sample, we also assume that the runtime is potentially available to an attacker. We assume that for all databases and for all x in the domain, the evaluations $g_D(x)$ take the same time to evaluate. As such, the runtime is proportional to the number of iterations in the sampler.

Theorem 2.2 gives a characterization of the privacy loss due to rejection sampling using (ϵ, δ)-DP.

Theorem 2.2. *Let $(\mathcal{D}, d)$ be a metric space of databases, and let T_D be the runtime of a rejection sampler for database D which has acceptance probability p_D. Note that $T_D \sim \text{Geom}(p_D)$. Call $R = \sup_{d(D,D') \leq 1} \frac{\log(1-p_D)}{\log(1-p_{D'})}$. The mechanism that releases the runtime T_D satisfies (ϵ, δ)-DP*

1. *for all $\epsilon \geq 0$ and $\delta(\epsilon) = (1 - 1/R) \exp\left(\frac{-\epsilon - \log(R)}{R-1}\right)$, or*
2. *for all $0 < \delta \leq (R-1)R^{R/(1-R)}$ and $\epsilon(\delta) = \log(1/R) + (R-1)(\log(1/\delta) + \log(1 - 1/R))$.*

Note that in Theorem 2.2, the runtime of a rejection sampler does not satisfy ϵ-DP for any finite ϵ, unless the probability of acceptance is constant across databases (i.e., $R = 1$).

2.1 Privacy risk of adaptive rejection sampling

We analyze the privacy risk of an idealized adaptive rejection sampler. Often adaptive rejection samplers update the proposal in a stochastic manner, based on the target value at previously rejected samples. In this section, we consider the setting where the proposal is updated in a deterministic manner, such as in Leydold et al. [20]. Unless the acceptance probabilities converge in a strong sense, an adaptive rejection sampler will not satisfy ϵ-DP for any finite ϵ.

Proposition 2.3. *Let $\{M_D \mid D \in \mathcal{D}\}$ be a privacy mechanism which satisfies ϵ_0-DP. Let $(p_i^D)_{i=1}^{\infty}$ be the sequence of acceptance probabilities for an adaptive rejection sampler for M_D. Call T_D the runtime of the adaptive sampler for M_D, with $pmf P(T_D = t) = p_t^D \prod_{i=1}^{n} (p_i^D)^{t-1}(1-p_i^D)$. Releasing a sample from M_D as well as the runtime T_D satisfies $(\epsilon_0 + \epsilon_T)$-DP, where ϵ_T satisfies $\epsilon_T \geq \log(p_t^D / p_t^{D'}) + \sum_{i=1}^{t-1} \log\left((1 - p_i^D)/(1 - p_i^{D'})\right)$, for all $t \geq 1$ and $d(D, D') \leq 1$.*

Proposition 2.3 shows that unless the acceptance probabilities are very closely related, it is not guaranteed that an adaptive rejection sampler will satisfy ϵ-DP for any finite ϵ.

3 Privacy-aware rejection samplers

The previous section showed that running a rejection sampler can result in an arbitrarily high privacy loss through the runtime. In this section we propose three modifications of the rejection sampling algorithm to ensure data-independent runtime. Finally, we apply our methods to develop a privacy-aware adaptive rejection sampler.

3.1 Constant runtime, truncated rejection sampling

One way to remove the privacy leak due to the runtime is to choose a number of iterations independent of the database. However, an accepted sample may not be found, and the probability of this event *does* depend on the database.

Of the methods we propose, the algorithm in Proposition 3.1 requires the weakest assumptions in that the only knowledge we require is a lower bound on the acceptance probability across the databases. However there is a small probability that no samples are accepted in the prescribed

number of iterations, which negatively impacts both the privacy and the utility of the mechanism.

Proposition 3.1. *Let $\{M_D \mid D \in \mathcal{D}\}$ be a family of mechanisms satisfying (ϵ_0, δ_0)-DP and (U_D, c_D) be such that $\widetilde{\pi}_D \le c_D U_D$ where $\widetilde{\pi}_D$ is an unnormalized density for M_D. Assume that $\alpha_0 \le 1/c_D \int \widetilde{\pi}_D(x)\,dx$ for all D, that is, α_0 is a lower bound on the acceptance probability in the rejection sampler across all databases. Given $\delta > 0$, run the sampler for $N = \frac{\log(1/\delta)}{\log(1/(1-\alpha_0))}$ iterations. If there is an accepted proposal, release the first one; if not, release an arbitrary output (such as one more draw from the proposal). Releasing the output as well as the runtime of this algorithm satisfies $(\epsilon_0, \delta_0 + \delta)$-DP.*

A benefit of the algorithm in Proposition 3.1 is that it can be vectorized and is embarrassingly parallelizable. Another benefit is that N grows only in the log of $1/\delta$. Roughly, by doubling the iterations, we can double the number of 'zeros' in the δ. The two major downsides are that the algorithm must be run much longer than a simple rejection sampler, and that it is not guaranteed that an accepted sample is found, which also reduces the utility. If no samples are accepted, then the output does not follow the correct distribution, introducing error in the sampling approximation.

3.2 Additive geometric wait time

In this section, we use the memoryless property of the geometric distribution to introduce an additive wait time based on a lower bound on the acceptance probability. The result is that the runtime of the algorithm is geometric with acceptance rate equal to the worst-case dataset.

The benefit of this method over the truncated rejection sampler is that a sample from the correct distribution is guaranteed, and the runtime is independent of the database. The downside is that the acceptance probability (or equivalently the integrating constant) for the present database is required.

Theorem 3.2. *Let $\{\pi_D \mid D\}$ be normalized target densities. Assume that for each π_D, we have normalized densities $U_D(x)$ as well as constants c_D such that for all x, $\pi_D(x) \le c_D U_D(x)$. Let $c \ge \sup_D c_D$. Consider the following scheme:*

1. *Run a rejection sampler, proposing from $U_D(x)$ and targeting $\pi_D(x)$ until acceptance*
2. *Call the accepted sample X. Also draw $Y \sim Unif(0,1)$.*
3. *If $Y < c_D/c$ return, else wait for $Geom(1/c)$ cycles.*

Then $X \sim \pi_D$, and the wait time follows $Geom(1/c)$, which does not depend on D.

3.3 Implicit wait-time via squeeze

We propose another method of producing an exact rejection sampler with data-independent runtime. Algorithm 1 avoids the need for the normalizing constant as in Theorem 3.2 by instead using a carefully tailored *squeeze* function.

Theorem 3.3. *Let $\{\widetilde{\pi}_D \mid D\}$ be (unnormalized) target densities. Assume that for each π_D, we have normalized densities $U_D(x)$ and $L_D(x)$ as well as constants $c_{L,D}$ and $c_{U,D}$ such that $c_{L,D}/c_{U,D}$ does not depend on D and such that for all x $c_{L,D} L_D(x) \le \widetilde{\pi}_D(x) \le c_{U,D} U_D(x)$. Then the output of Algorithm 1 with $\widetilde{\pi} = \widetilde{\pi}_D$, $U = U_D$, $L = L_D$, $c_U = c_{U,D}$, $c_L = c_{L,D}$ has distribution π_D and runtime $\mathrm{Geom}(c_{L,D}/c_{U,D})$, which does not depend on D.*

Algorithm 1 Privacy-aware rejection sampling via squeeze

INPUT: $\widetilde{\pi}, U, L, c_U$, and c_L such that $c_L L(x) \le \widetilde{\pi}(x) \le c_U U(x)$ for all x

Set anyAccepted=FALSE
Sample $X \sim U(x)$
Sample $Y \sim \mathrm{Unif}(0,1)$
if $Y \le \frac{\widetilde{\pi}(X)}{c_U U(X)}$ and anyAccepted==FALSE **then**
    Set $X_s = X$
    Set anyAccepted=TRUE
end if
if $Y \le \frac{c_L L(X)}{c_U U(X)}$ **then**
    Return X_s
else
    Go to 2.
end if

OUTPUT: X_s

While the assumption of the squeeze functions in Theorem 3.3 may seem odd, it is in fact strictly weaker than knowing the integrating constant for $\widetilde{\pi}_D$, as was required in Section 3.2, as shown in Proposition 3.4.

Proposition 3.4. *Let $\{\pi_D \mid D\}$ be normalized target densities. Assume that for each π_D, we have normalized densities $U_D(x)$ and constants $c_{U,D}$ such that $\pi_D(x) \le c_{U,D} U_D(x)$. Let $c \ge \sup_D c_{U,D}$. Then the squeeze function $L_D = \pi_D$, with constant $c_{L,D} = c_{U,D}/c$ satisfies the assumptions of Theorem 3.3.*

3.4 Adaptive rejection sampler for log-Holder

In this section, we use the squeeze method of Section 3.3 to develop an adaptive rejection sampler with data-independent runtime for log-Holder densities. Our method is a modification of the (nearly) minimax optimal sampler of Achddou et al. [1]. Let $\pi_D(x) \propto \exp(g_D(x))$ be an unnormalized target density on a bounded convex set C, where g_D is (s, H)-Holder for all datasets D: $|g_D(x) - g_D(y)| \le H\|x - y\|^s$ for all D and for all $x, y \in C$.

Theorem 3.5. *Let $\{\widetilde{\pi}_D = \exp(g_D) \mid D\}$ be unnormalized target densities which have support on a bounded convex set C. Suppose that for all D, g_D is (s, H)-Holder with norm $\|\cdot\|$ on C. Then Algorithm 2 results in N i.i.d. samples from $\widetilde{\pi}$ and has running time which does not depend on D.*

As in Achddou et al. [1], we can make the adaptive sampler much easier to implement by considering the following special case of Algorithm 2: 1) use the ℓ_∞ norm in the Holder definition, 2) set $C = [0,1]^d$, 3) approximate the nearest neighbor calculation $P_T(y)$ on a grid, as described in Achddou et al. [1, Definition 4]. These modifications make the construction, evaluation, and sampling of the proposal $\exp(\hat{g})$ computationally efficient, even in high dimensions. The adapt-reject

Algorithm 2 Privacy-aware adaptive rejection

INPUT: g an (s, H)-Holder function on a bounded convex set $C \subset \mathbb{R}^d$ for some norm $\|\cdot\|$, initial evaluations $\{(x_1, g(x_1)), \ldots, (x_n, g(x_n))\}$, and a "nearest neighbor" map $P_T(\cdot) : C \to T$ for any finite set $T \subset C$, the number N of i.i.d. samples desired from $\pi(x) \propto \exp(g(x))I(x \in C)$

1: Set `prevAccepted=FALSE`
2: Set `numSamples=0`
3: Set `publishedSamples=` $\emptyset$
4: Set $S = \{(x_1, g(x_1)), \ldots, (x_n, g(x_n))\}$
5: Set $T = \{x \mid (x, y) \in S \text{ for some } y\}$
6: **while** `numSamples<` N **do**
7: Define $\hat{g}(y) = g(P_T(y))$ for all $y \in C$ (only evaluations of g from S)
8: Set $\hat{r} \geq \sup_{y \in C} H\|y - P_T(y)\|^s$
9: Sample $Y \sim \exp(\hat{g}(y))/(\int_C \exp(\hat{g}(y))\, dy)$
10: Sample $U \sim \text{Unif}(0, 1)$
11: **if** $U \leq \exp(g(Y))/\exp(\hat{g}(Y) + \hat{r})$ and `anyAccepted=FALSE` **then**
12: Set $X_s = Y$
13: Set `anyAccepted=TRUE`
14: **end if**
15: **if** $U \leq \exp(-2\hat{r})$ **then**
16: Publish X_s and append X_s to `publishedSamples`
17: Increment `numSamples` by 1
18: Set `anyAccepted=FALSE`
19: **end if**
20: Choose $Z \in C \setminus T$ based on only T, H and s
21: Append $(Z, g(Z))$ to S
22: Append Z to T
23: **end while**

OUTPUT: `publishedSamples`, which can be published in a stream

(lines 9-19) and the update steps (lines 20-22) can also be done in batches to avoid updating the function $\hat{g}$ too often.

Remark 3.6. There are several prior DP works on the exponential mechanism, where the utility function is assumed to be Holder, and where Algorithm 2 can be applied. Minami et al. [23] assume Lipschitz and concave utility functions. Bassily et al. [6] and Bassily et al. [7] derive optimal DP mechanisms under the assumption of Lipschitz and convex empirical risk objective functions, as well as a bounded domain, which result in implementations of the exponential mechanism. In part of their work, Ganesh and Talwar [12] assume Lipschitz and L-smooth utility functions in the exponential mechanism. Wang et al. [29] study non-convex empirical risk minimization problems with objectives that are Lipschitz and M-smooth.

4 Exponential mechanism sampling

In this section, we explore some some instances of the exponential mechansim that satisfy the assumptions of the rejection samplers proposed in Section 3, and so allow for a privacy-preserving implementation.

4.1 Strongly concave and L-smooth log-density

We consider instances of the exponential mechanism where the utility function g_D is both strongly concave and L-smooth. These are the same properties that Ganesh and Talwar [12] assume. Both Awan et al. [4] and Minami et al. [23] assume strongly concave utility functions in the exponential mechanism. Other private empirical risk minimization works also commonly assume L-smooth and strong convexity [6, 7, 18].

Under the strongly concave and L-smooth assumptions, we are able to derive upper and lower bounds for the target, which satisfy the requirements of Theorem 3.3.

Lemma 4.1. *Let $M(x) \propto \exp(g(x))$ be the target density, where $g : \mathbb{R}^d \to \mathbb{R}$ is twice-differentiable, α-strongly concave, and L-smooth. Call $x^* := \arg\max_x g(x)$. Using $\phi_d(x; \mu, \Sigma)$ to denote the pdf of $N_d(\mu, \Sigma)$. Then,*

$$\exp(g(x^*))\,(2\pi/L)^{d/2}\,\phi_d(x; x^*, (1/L)I)$$
$$\leq \exp(g(x)) \leq \exp(g(x^*))\,(2\pi/\alpha)^{d/2}\,\phi_d(x; x^*, (1/\alpha)I).$$

Given the bounds in Lemma 4.1, we can now implement the squeeze-function rejection sampler of Section 3.3. Note that the acceptance probability when targeting the lower bound is $(\alpha/L)^{d/2}$, which does not depend on D.

4.2 KNG/Gradient mechanism

An alternative to the exponential mechanism is the K-norm gradient mechanism (KNG), proposed in Reimherr and Awan [26], also known as the gradient mechanism [2]. KNG has been applied to applications such as geometric median estimation, and linear and quantile regression [2, 26]. Given an objective function $g_D(x)$, KNG samples from $M_D(x) \propto \exp(-\frac{\epsilon}{2\Delta}\|\nabla g_D(x)\|_K)$, where $\Delta \geq \|\nabla g_D(x) - g_{D'}(x)\|_K$, for all x and $d(D, D) \leq 1$.

While the exponential mechanism with a strongly concave utility is naturally approximated by a Gaussian distribution [4], KNG is closely related to the K-norm distributions [26]. The K-norm mechanism was introduced in Hardt and Talwar [14], and were also studied in Awan and Slavković [5].

Definition 4.2 ([14])**.** Let $\|\cdot\|_K$ be a norm on $\mathbb{R}^d$, with associated unit norm ball: $K = \{x \in \mathbb{R}^D \mid \|x\|_K \leq 1\}$. The K-norm distribution with location m and scale s has density $f(x; m, s) = c^{-1} \exp\left(\frac{-1}{s}\|x - m\|_K\right)$, where $c = (d!)s^d \text{Vol}(K)$.

Under similar assumptions as those in Reimherr and Awan [26, Theorem 3.1], Lemma 4.3 gives upper and lower bounds which satisfy the assumptions required for Theorem 3.3.

Lemma 4.3. *Let $M(x) \propto \exp(-\|\nabla g_D(x)\|_2)$ be the target, where $g_D : \mathbb{R}^d \to \mathbb{R}$ is twice-differentiable, α-strongly convex, and L-smooth. Call $x^* := \arg\min_x g_D(x)$. Write $\psi_d(x; m, s)$ to denote the pdf of a K-norm distribution with location m, scale s, and ℓ_2 norm. Denote $\text{Vol}_d(\ell_2) = \frac{2^d \Gamma^d(1+1/2)}{\Gamma(1+d/2)}$ the volume of the unit ℓ_2 ball in $\mathbb{R}^d$. Then,*

$$(d!)L^{-d}\text{Vol}_d(\ell_2)\psi_d(x; x^*, 1/L)$$
$$\leq \exp(-\|\nabla g_D(x)\|_2) \leq (d!)\alpha^{-d}\text{Vol}_d(\ell_2)\psi_d(x; x^*, 1/\alpha).$$

Lemma 4.3 gives bounds that can be used to implement Algorithm 1. The acceptance probability when targeting the lower bound is $(\alpha/L)^d$, which is independent of D.

If the underlying utility in KNG is L-smooth, then the log-density is L-Lipschitz. As such, we can apply the adaptive rejection sampler of Section 3.4.

References

[1] Juliette Achddou, Joseph Lam-Weil, Alexandra Carpentier, and Gilles Blanchard. 2019. A minimax near-optimal algorithm for adaptive rejection sampling. In *Algorithmic Learning Theory*. PMLR, 94–126.

[2] Hilal Asi and John C Duchi. 2020. Instance-optimality in differential privacy via approximate inverse sensitivity mechanisms. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 14106–14117. https://proceedings.neurips.cc/paper/2020/file/a267f936e54d7c10a2bb70dbe6ad7a89-Paper.pdf

[3] Hilal Asi and John C Duchi. 2020. Near instance-optimality in differential privacy. *arXiv preprint arXiv:2005.10630* (2020).

[4] Jordan Awan, Ana Kenney, Matthew Reimherr, and Aleksandra Slavković. 2019. Benefits and pitfalls of the exponential mechanism with applications to hilbert spaces and functional pca. In *International Conference on Machine Learning*. PMLR, 374–384.

[5] Jordan Awan and Aleksandra Slavković. 2020. Structure and sensitivity in differential privacy: Comparing k-norm mechanisms. *J. Amer. Statist. Assoc.* (2020), 1–20.

[6] Raef Bassily, Adam Smith, and Abhradeep Thakurta. 2014. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. IEEE, 464–473.

[7] Raef Bassily, Adam Smith, and Abhradeep Thakurta. 2014. Private empirical risk minimization, revisited. *rem* 3 (2014), 19.

[8] Kamalika Chaudhuri, Anand D Sarwate, and Kaushik Sinha. 2013. A Near-Optimal Algorithm for Differentially-Private Principal Components. *Journal of Machine Learning Research* 14 (2013).

[9] Christos Dimitrakakis, Blaine Nelson, Zuhe Zhang, Aikateirni Mitrokotsa, and Benjamin Rubinstein. 2017. Differential privacy for Bayesian inference through posterior sampling. *Journal of machine learning research* 18, 11 (2017), 1–39.

[10] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[11] Dan Feldman, Amos Fiat, Haim Kaplan, and Kobbi Nissim. 2009. Private coresets. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*. 361–370.

[12] Arun Ganesh and Kunal Talwar. 2020. Faster Differentially Private Samplers via Rényi Divergence Analysis of Discretized Langevin MCMC. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 7222–7233. https://proceedings.neurips.cc/paper/2020/file/50cf0fe63e0ff857e1c9d01d827267ca-Paper.pdf

[13] Andreas Haeberlen, Benjamin C Pierce, and Arjun Narayan. 2011. Differential Privacy Under Fire.. In *USENIX Security Symposium*, Vol. 33.

[14] Moritz Hardt and Kunal Talwar. 2010. On the geometry of differential privacy. In *Proceedings of the forty-second ACM symposium on Theory of computing*. 705–714.

[15] Anco Hundepool, Josep Domingo-Ferrer, Luisa Franconi, Sarah Giessing, Eric Schulte Nordholt, Keith Spicer, and Peter-Paul De Wolf. 2012. *Statistical disclosure control*. John Wiley & Sons.

[16] G Joy Persial, M Prabhu, and R Shanmugalakshmi. 2011. Side channel attack-survey. *Int J Adva Sci Res Rev* 1, 4 (2011), 54–57.

[17] Michael Kapralov and Kunal Talwar. 2013. On differentially private low rank approximation. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 1395–1414.

[18] Daniel Kifer, Adam Smith, and Abhradeep Thakurta. 2012. Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*. JMLR Workshop and Conference Proceedings, 25–1.

[19] Julia Lane, Victoria Stodden, Stefan Bender, and Helen Nissenbaum. 2014. *Privacy, big data, and the public good: Frameworks for engagement*. Cambridge University Press.

[20] Josef Leydold, Erich Janka, and Wolfgang Hörmann. 2002. Variants of transformed density rejection and correlation induction. In *Monte Carlo and Quasi-Monte Carlo Methods 2000*. Springer, 345–356.

[21] Frank McSherry and Kunal Talwar. 2007. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*. IEEE, 94–103.

[22] Frank D McSherry. 2009. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. 19–30.

[23] Kentaro Minami, Hiromi Arai, Issei Sato, and Hiroshi Nakagawa. 2016. Differential privacy without sensitivity. In *Advances in Neural Information Processing Systems*. 956–964.

[24] Prashanth Mohan, Abhradeep Thakurta, Elaine Shi, Dawn Song, and David Culler. 2012. GUPT: privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. 349–360.

[25] Shirin Nilizadeh, Yannic Noller, and Corina S Pasareanu. 2019. DiffFuzz: differential fuzzing for side-channel analysis. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 176–187.

[26] Matthew Reimherr and Jordan Awan. 2019. Kng: The k-norm gradient mechanism. In *Advances in Neural Information Processing Systems*. 10208–10219.

[27] Indrajit Roy, Srinath TV Setty, Ann Kilzer, Vitaly Shmatikov, and Emmett Witchel. 2010. Airavat: Security and privacy for MapReduce.. In *NSDI*, Vol. 10. 297–312.

[28] Joshua Snoke and Aleksandra Slavković. 2018. pMSE mechanism: differentially private synthetic data with maximal distributional similarity. In *International Conference on Privacy in Statistical Databases*. Springer, 138–159.

[29] Di Wang, Changyou Chen, and Jinhui Xu. 2019. Differentially private empirical risk minimization with non-convex loss functions. In *International Conference on Machine Learning*. PMLR, 6526–6535.

[30] Yu-Xiang Wang, Stephen Fienberg, and Alex Smola. 2015. Privacy for free: Posterior sampling and stochastic gradient monte carlo. In *International Conference on Machine Learning*. 2493–2502.

[31] Larry Wasserman and Shuheng Zhou. 2010. A statistical framework for differential privacy. *J. Amer. Statist. Assoc.* 105, 489 (2010), 375–389.

[32] Zuhe Zhang, Benjamin IP Rubinstein, and Christos Dimitrakakis. 2016. On the differential privacy of Bayesian inference. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. 2365–2371.