Journal of Machine Learning Research 24 (2023) 1-32 Submitted 8/21; Revised 9/22; Published 1/23

Privacy-Aware Rejection Sampling

Jordan Awan JAWAN@PURDUE.EDU
Department of Statistics
Purdue University
West Lafayette, IN 47907, USA

Vinayak Rao VARAO@PURDUE.EDU
Department of Statistics
Purdue University
West Lafayette, IN 47907, USA

Editor: Adrian Weller

Abstract

While differential privacy (DP) offers strong theoretical privacy guarantees, implementations of DP mechanisms may be vulnerable to side-channel attacks, such as timing attacks. When sampling methods such as MCMC or rejection sampling are used to implement a privacy mechanism, the runtime can leak private information. We characterize the additional privacy cost due to the runtime of a rejection sampler in terms of both (ϵ, δ)-DP as well as f-DP. We also show that unless the acceptance probability is constant across databases, the runtime of a rejection sampler does not satisfy ϵ-DP for any ϵ. We show that there is a similar breakdown in privacy with adaptive rejection samplers. We propose three modifications to the rejection sampling algorithm, with varying assumptions, to protect against timing attacks by making the runtime independent of the data. The modification with the weakest assumptions is an approximate sampler, introducing a small increase in the privacy cost, whereas the other modifications give perfect samplers. We also use our techniques to develop an adaptive rejection sampler for log-Hölder densities, which also has data-independent runtime. We give several examples of DP mechanisms that fit the assumptions of our methods and can thus be implemented using our samplers.

Keywords: differential privacy, side-channel, timing attack, perfect sampler, exponential mechanism

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 (Lane et al., 2014). While there have been many methods of statistical disclosure control to combat this problem (Hundepool et al., 2012), differential privacy (DP) (Dwork et al., 2006) has arisen as the state-of-the-art framework for privacy protection, and is currently being implemented by Google (Erlingsson et al., 2014), Apple (Tang et al., 2017), Microsoft (Ding et al., 2017), and the US Census (Abowd, 2018). 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 (Wasserman and Zhou, 2010).

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, and may consist of the execution time, power consumption, or memory usage of the system, to name a few (Joy Persial et al., 2011; Nilizadeh et al., 2019). 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. Such a side-channel may be present if the DP mechanism is part of a query-response framework, where users submit queries and the curator replies with a DP response; in this model, the adversary may measure the time between submitting the query and receiving the answer, and use this information as part of their attack. PINQ (McSherry, 2009) and Airavat (Roy et al., 2010) were two of the earliest DP implementations, but were shown by Haeberlen et al. (2011) to be vulnerable to timing attacks. FUZZ (Haeberlen et al., 2011) and GUPT (Mohan et al., 2012) 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, but is nontrivial for complex DP mechanisms.

One of the most common and powerful DP mechanisms is the exponential mechanism (McSherry and Talwar, 2007) which results in an unnormalized density of the form $\exp(g_D(x))$ that must be sampled from, where g_D is some function that depends on the database D. The exponential mechanism has been widely used to tackle problems such as principal component analysis (Chaudhuri et al., 2013; Kapralov and Talwar, 2013; Awan et al., 2019), K-means clustering, (Feldman et al., 2009), convex optimization (Bassily et al., 2014a,b), robust regression (Asi and Duchi, 2020b), linear and quantile regression (Reimherr and Awan, 2019), synthetic data (Snoke and Slavković, 2018), and Bayesian data analysis (Wang et al., 2015; Minami et al., 2016; Zhang et al., 2016; Dimitrakakis et al., 2017) to name a few.

A challenge however is that for functions $g_D(x)$ encountered in practice, the unnormalized density $\exp(g_D(x))$ is often difficult to sample from. In statistics and machine learning, there are many computational techniques to produce either exact or approximate samples from such distributions, including Markov chain Monte Carlo (MCMC), rejection sampling, and approximate Bayesian computing. 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 runtime of the algorithm depends on the database, then this side-channel may leak private information (Haeberlen et al., 2011).

We will consider the runtime of the algorithm as an additional output accessible to an adversary, and we will require that both the official output and the runtime jointly satisfy differential privacy. As Haeberlen et al. (2011) point out, the simplest solution is to make the runtime independent of the dataset. In this paper we propose different modifications, under different assumptions, which produce rejection samplers with data-independent runtime, and are thus immune to timing attacks.

Contributions First, we quantify the privacy risk of rejection and adaptive rejection sampling without any privacy-preserving modifications. As a properly implemented 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 due to the runtime of a simple rejection sampler in terms of both (ϵ,δ)-DP and f-DP (Dong et al., 2022). We also show that the runtime of a simple rejection sampler does not satisfy ϵ-DP for any finite ϵ unless the acceptance rate is constant across databases. We similarly show that the runtime of an adaptive rejection sampler does not satisfy ϵ-DP unless acceptance probabilities across databases converge in terms of a certain series.

Given the increased privacy risk due to the runtime, we propose several modifications to rejection samplers, which make the runtime independent of the database: 1) choose the number of iterations to run the sampler ahead of time, based on a lower bound on the acceptance probability, 2) introduce an additive wait-time based on a worst-case dataset, 3) use squeeze functions to add an implicit wait-time. We also propose an adaptive rejection sampler with data-independent runtime, which can be applied to any log-Hölder density. The adaptive sampler is a modification of the (nearly) minimax optimal sampler from Achddou et al. (2019), using the technique of squeeze functions. Finally, we give examples of the exponential mechanism which satisfy the assumptions of our methods.

Related work Often side-channels are handled using more relaxed metrics than DP, such as min-entropy (Smith, 2009). However, the point of view of this paper is that if the dataset in question is judged to require the protection of differential privacy, then we must ensure that *all channels* are protected in the DP framework. Thus, while for other applications it may be appropriate to use a weaker protection for side channels, in DP applications, the runtime must also satisfy DP. See Haeberlen et al. (2011) for a similar discussion.

Besides timing side-channels, there are other notable side-channel attacks that have been effective against DP implementations. Haeberlen et al. (2011) showed that when the privacy budget is chosen based on the database, that the budget is another side-channel. Wagh et al. (2018) consider the privacy cost of RAM access, and propose a differential privacy regime to formally protect the RAM access. Dodis et al. (2012) and Garfinkel and Leclerc (2020) explore the concerns of using pseudo-random number generators in the implementation of DP systems. Mironov (2012) showed that when implementing DP mechanisms with floating point arithmetic, privacy can be arbitrarily compromised by the artifacts in the least significant bit. Ilvento (2020) provide an implementation of the exponential mechanism on finite state spaces that is immune to the floating point attacks, but which is admitted to be susceptible to timing attacks.

A different approach to sampling the exponential mechanism is using MCMC techniques, and there have been some prior works characterizing the additional privacy cost of these approximate samplers. Usually convergence of MCMC methods is characterized in terms of total variation distance, and Minami et al. (2016) showed that these guarantees can be imported to produce approximate DP samples with an increased 'delta' in a fixed number of iterations. Ganesh and Talwar (2020) expanded upon the results of Vempala and Wibisono (2019) to show that Langevin MCMC converges in Rényi divergence, which allows for the quantification of the privacy loss by sampling in terms of Rényi DP. Rényi divergences are much stronger than total variation, and have been used in various definitions of DP (Mironov, 2017; Bun and Steinke, 2016; Bun et al., 2018). Minami et al.

(2016) also study Langevin MCMC, but characterize the privacy cost in terms of (ϵ, δ)-DP. Seeman et al. (2021) develop an exact sampler for the exponential mechanism based on an MCMC procedure with artificial atoms, however, they acknowledge that their approach does not protect against timing side-channels. To our knowledge, there has been no prior work quantifying the privacy risk of rejection sampling, or proposing rejection samplers with data-independent runtime.

2. Background and notation

In this section, we review the necessary background on differential privacy and rejection sampling. We also set the notation for the rest of the paper.

Let X and Y be random variables on a measurable space $(\mathscr{Y}, \mathscr{F})$, with corresponding probability measures μ_X and μ_Y. The *max-divergence* of Y with respect to X is $D_\infty(Y||X) = \sup_{B\in\mathscr{F}} \log\left(\frac{\mu_Y(B)}{\mu_X(B)}\right)$. If μ_X dominates μ_Y, then $D_\infty(Y||X) = \sup_{y\in\mathscr{Y}} \log \frac{d\mu_Y}{d\mu_X}(y)$, where $\frac{d\mu_Y}{d\mu_X}$ is the Radon-Nikodym derivative of μ_Y with respect to μ_X. The *symmetric max-divergence* is $D^S_\infty(X, Y) := \max\{D_\infty(X||Y), D_\infty(Y||X)\}$.

For a distribution M, we typically write $\widetilde{\pi}(x)$ for an unnormalized density of M, $\pi(x) = \widetilde{\pi}(x)/\int \widetilde{\pi}(x)\, dx$, and $g = \log(\widetilde{\pi})$ (equivalently, $\widetilde{\pi}(x) = \exp(g(x))$. We write $U(x)$ to denote a density that upper bounds $\widetilde{\pi}$ as $\widetilde{\pi}(x) \leq c_U U(x)$ for some constant c_U. Similarly, we write $L(x)$ for a density that lower bounds $\widetilde{\pi}$ as $c_L L(x) \leq \widetilde{\pi}(x)$ for a constant c_L. In rejection sampling, U is called the *proposal distribution*, and L is the *squeeze function.*

2.1 Differential privacy

Differential privacy (DP), introduced in Dwork et al. (2006), is a framework to characterize the privacy risk of a given algorithm, and offers techniques to design mechanisms which limit privacy loss. DP methods require the introduction of additional randomness, beyond sampling, in order to offer a notion of plausible deniability. Given the output of a DP mechanism, it is difficult for an adversary to determine whether a particular individual participated in the dataset or not. While an idealized algorithm may be proven to be differentially private, to characterize the actual privacy cost of a given implementation, one must consider all side-channels such as the runtime as part of the DP output (Haeberlen et al., 2011).

Definition 1 (Privacy mechanism) *Given a metric space $(\mathscr{D}, d)$, which represents the set of possible databases, a set of probability measures $\{M_D \mid D \in \mathscr{D}\}$ on a common space $\mathscr{Y}$ is called a* privacy mechanism*.*

The space $\mathscr{D}$ represents the space of possible databases, and it is common to take $\mathscr{D} = \mathscr{X}^n$ for some set $\mathscr{X}$, with $\mathscr{X}$ representing the possible contributions of one individual in the database. In that case, the metric d is often chosen to be the Hamming distance, so that $d(D, D') \leq 1$ represents that D and D' are *adjacent* databases, differing in only one individual's contribution.

When implementing a privacy mechanism, we publish one sample from M_D, which satisfies some form of privacy.

Definition 2 **((ϵ,δ)-DP:Dwork et al., 2006)** *Given a metric space $(\mathscr{D}, d)$, $\epsilon \geq 0$ and $\delta \in [0,1]$, a privacy mechanism $\{M_D\}$ on the space $\mathscr{Y}$ satisfies (ϵ,δ)-differential privacy if for all measurable sets $B \in \mathscr{Y}$ 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. Typically, ϵ is chosen to be a small constant such as 1 or 0.1, whereas usually $\delta \ll 1/n$. In the case where $\delta = 0$, we call $(\epsilon, 0)$-DP "pure differential privacy," and write ϵ-DP. A privacy mechanism satisfying ϵ-DP is equivalent to requiring that $D^S_\infty(M_D || M_{D'}) \leq \epsilon$ for all $d(D,D') \leq 1$, where D^S_∞ is the symmetric max-divergence.

While we phrase most of our results in terms of (ϵ,δ)-DP, another useful formulation of DP is f-DP (Dong et al., 2022), which is expressed in terms of hypothesis tests. f-DP is based on bounding the receiver-operator curve (ROC) or tradeoff function when testing between two adjacent databases, given the output of a privacy mechanism. For two probability distributions P and Q, the *tradeoff function* is the smallest type-II error as a function of the type-I error. Formally, the *tradeoff function* for P and Q is $T(P,Q) : [0,1] \to [0,1]$, which is defined as $T(P,Q)(\alpha) = \inf_\phi \{1 - \mathbb{E}_Q(\phi) \mid \mathbb{E}_P(\phi) \leq \alpha\}$, where the infinimum is over all possible tests ϕ. Being equivalent to ROC, the tradeoff function captures the difficulty of distinguishing between P and Q. A function $f : [0,1] \to [0,1]$ is a tradeoff function if and only if f is convex, continuous, decreasing, and $f(x) \leq 1 - x$ for all $x \in [0,1]$ (Dong et al., 2022, Proposition 1).

Definition 3 **(f-DP: Dong et al., 2022)** *Let f be a tradeoff function. A privacy mechanism M on the metric space $(\mathscr{D}, d)$ satisfies f-DP if*

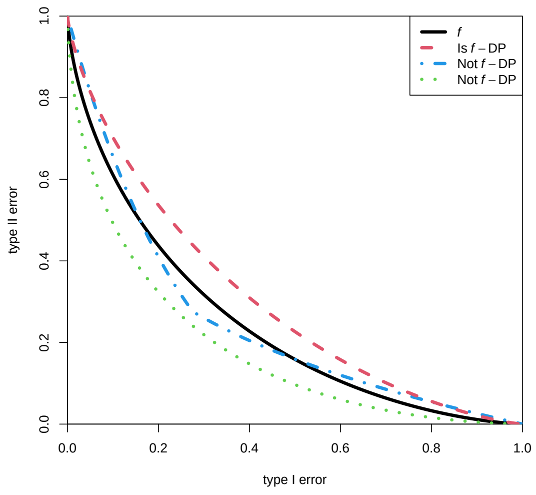
$$T(M_D, M_{D'})(\alpha) \geq f(\alpha) \quad \forall \alpha \in [0,1],$$

for all $D, D' \in \mathscr{D}$ such that $d(D,D') \leq 1$.

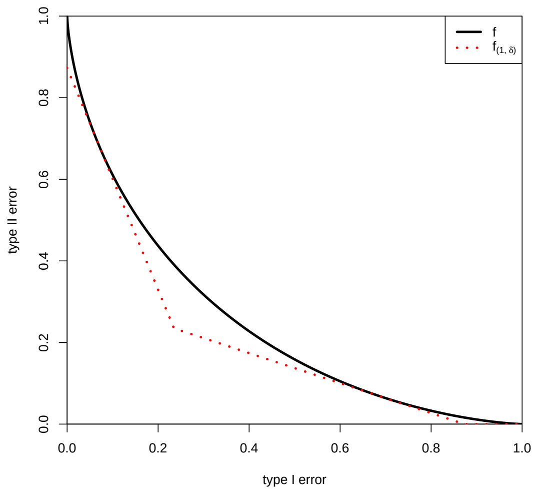
See Figure 1a for examples of tradeoff functions which do and do not satisfy f-DP for a particular f. Without loss of generality we can assume that f is symmetric: $f(\alpha) = f^{-1}(\alpha)$, where $f^{-1}(\alpha) = \inf\{t \in [0,1] \mid f(t) \leq \alpha\}$. This is due to Dong et al. (2022, Proposition 2), which states that for a given f and a mechanism M that is f-DP, there exists a symmetric $f^* \geq f$ such that M is f^*-DP.

It turns out that (ϵ,δ)-DP is a special case of f-DP, where f is taken to be a particular piecewise linear function. Specifically, let $\epsilon \geq 0$ and $\delta \in [0,1]$, and define $f_{\epsilon,\delta}(\alpha) = \max\{0, 1-\delta-\exp(\epsilon)\alpha, \exp(-\epsilon)(1-\delta-\alpha)\}$. Then a privacy mechanism M satisfies (ϵ,δ)-DP if and only if it satisfies $f_{\epsilon,\delta}$-DP (Dong et al., 2022, Proposition 3). The following proposition, based on Dong et al. (2022, Propositions 5 and 6), gives a simple conversion between f-DP and (ϵ,δ)-DP, by determining the linear functions which lower bound f.

Proposition 4 *Let f be a symmetric tradeoff function. If a privacy mechanism satisfies f-DP, then it satisfies (ϵ,δ)-DP provided that $(1-\delta) - \exp(\epsilon)\alpha \leq f(\alpha)$ for all $\alpha \in [0,1]$.*



(a) A plot of three examples of $T(M(D), M(D'))$. Only the red, dashed tradeoff curve satisfies f-DP.



(b) A tradeoff function, as well as its conversion to $(1, \delta)$-DP, where $\delta \approx 0.127$.

Figure 1: Examples of tradeoff functions, and the relation between f-DP and (ϵ, δ)-DP.

Proof We need to show that $f_{\epsilon,\delta}(\alpha) \leq f(\alpha)$ for all $\alpha \in [0, 1]$. By symmetry of f and $f_{\epsilon,\delta}$, the condition stated is sufficient. ■

If the tradeoff function f makes the inequalities of Definition 3 tight, then by Proposition 4 the tightest (ϵ, δ)-DP guarantee takes a tangent line of f and sets $(1 - \delta)$ to be the y-intercept and $-\exp(\epsilon)$ to be its slope. This approach gives a precise conversion from f-DP to (ϵ, δ)-DP, which we use in Theorem 10. In fact, there is a stronger duality between f-DP and a family of $(\epsilon, \delta(\epsilon))$-DP characterizations, described in Dong et al. (2022, Propositions 5 and 6). Figure 1b illustrates the conversion from f-DP to (ϵ, δ)-DP.

An important property of both (ϵ, δ)-DP and f-DP is that it is robust to *post-processing*. That is, if a privacy mechanism satisfies DP, then applying any deterministic or randomized algorithm to the output cannot degrade the DP guarantee. This property is related to data processing inequalities.

Proposition 5 (Post-processing: Dwork et al., 2014; Dong et al., 2022) *Let M be a privacy mechanism with output space $\mathscr{Y}$, f a tradeoff function, $\epsilon \geq 0$, and $\delta \in [0, 1]$. Let* Proc *be a potentially randomized mapping from $\mathscr{Y}$ to $\mathscr{Z}$. Then*

1. *if M satisfies (ϵ, δ)-DP, then* $\mathrm{Proc} \circ M$ *satisfies (ϵ, δ)-DP;*

2. *if M satisfies f-DP, then* $\mathrm{Proc} \circ M$ *satisfies f-DP.*

2.2 Exponential mechanism

Having established the definitions of both (ϵ, δ)-DP and f-DP, there remains the question of how to construct a privacy mechanism for a given statistical task. A general and powerful technique, and one that will be the focus of this paper, is the *exponential mechanism* (McSherry and Talwar, 2007). Given a utility function g_D, where large values of g_D indicate higher utility, the exponential mechanism samples from the unnormalized density $\widetilde{\pi}_D(x) = \exp(g_D(x))\pi_0(x)$, where π_0 is a base measure. This mechanism satisfies $(2/\Delta, 0)$-DP where Δ is the *sensitivity* of g_D:

$$\Delta \geq \sup_{d(D,D')\leq 1} \sup_x |g_D(x) - g_{D'}(x)|.$$

Often π_0 is chosen to be Lebesgue measure, but it can also be chosen to be a probability measure similar to a prior (Wang et al., 2015; Minami et al., 2016; Dimitrakakis et al., 2017). In infinite-dimensional function spaces, there is no translation-invariant measure, so a nontrivial base measure must be used (Awan et al., 2019). Many statistical tasks can be expressed as finding the solution to a minimization or maximization problem of some objective function (e.g., log-likelihood function, sum of squared error, or a general empirical risk function). For these tasks, it is natural to choose the utility function in the exponential mechanism to be some transformation of such an objective function. For example, Reimherr and Awan (2019) show that when an objective function $\xi_D(x)$ is strongly convex, sampling from the exponential mechanism with utility function $g(x) = -\|\nabla \xi_D(x)\|$ results in an estimator which satisfies $x^* = \arg\min_x \xi_D(x) + O_p(n^{-1})$. Though the exponential mechanism was designed with $(\epsilon, 0)$-DP in mind, it has been shown that when the utility function satisfies additional assumptions such as concavity, Lipschitz continuity, or strong concavity, the exponential mechanism may satisfy (ϵ, δ)-DP (Minami et al., 2016; Dimitrakakis et al., 2017), even when the sensitivity Δ is infinite.

While the exponential mechanism is very flexible and offers high utility guarantees, sampling $\exp(g_D(x))$ exactly is generally very challenging. While specific implementations of the exponential mechanism sometimes have efficient sampling schemes (e.g., Bassily et al., 2014a,b; Asi and Duchi, 2020a,b), in general, more sophisticated computational sampling techniques are needed. For example, Chaudhuri et al. (2012, 2013) and Awan et al. (2019) use a Gibbs sampler to implement the exponential mechanism in the application of principal component analysis, using heuristics to argue convergence. Reimherr and Awan (2019) use MCMC implementations of their proposed K-norm gradient (KNG) mechanism, but leave considerations of the cost of the implementation for future work. Snoke and Slavković (2018) propose an instance of the exponential mechanism for synthetic data, which they sample using the Metropolis algorithm, without considering the privacy cost of the sampler.

2.3 Rejection sampling

Given the structure of the unnormalized density, sampling from $\exp(g_D(x))$ is often well suited to rejection sampling. Given an unnormalized target density $\pi(x) \propto \widetilde{\pi}(x) = \exp(g(x))$, which is difficult to sample from, and a simpler *proposal* density $U(x)$ which satisfies $\widetilde{\pi}(x) \leq cU(x)$ for some c and all x, a rejection sampler draws $X \sim U(x)$ and accepts the sample with probability $\widetilde{\pi}(X)/(cU(X))$. This process is repeated until a sample is accepted, and it is easy to show that the accepted sample is distributed as $X \sim \pi(x)$. The requirements to implement a rejection sampler are that we can evaluate $\widetilde{\pi}(x)$, and determine $U(x)$ and c which satisfy the above inequality. We will call these samplers *simple rejection* *samplers* when we need to distinguish these from adaptive rejection samplers, which we introduce later in this section. See Martino (2018) for an extensive introduction to rejection samplers.

The marginal probability of accepting a sample at any particular iteration from a simple rejection sampler is $p = c^{-1} \int \widetilde{\pi}(x)\ dx$, so that the number of iterations T needed to obtain an accepted sample follows a geometric distribution: $T \sim \text{Geom}(p)$. In this paper we assume that the geometric distribution has support $1, 2, 3, \ldots$, with pmf $P(T = k) = (1-p)^{k-1}(p)$ for $k = 1, 2, 3, \ldots$.

While rejection samplers allow exact samples to be drawn from an intractable target distribution, the acceptance probability p typically decays exponentially with dimension, making them suitable only for low-dimensional problems. Adaptive rejection samplers attempt to address this shortcoming, and proceed by producing a *sequence* of upper bounds $U_n(x)$ and constants c_n such that $\widetilde{\pi}(x) \leq c_n U_n(x)$ and such that the acceptance probability increases with n. Just like a simple rejection sampler, conditional on acceptance, adaptive rejection samplers produce samples $X \sim \pi(x)$. Typically, the upper bounds are updated stochastically, using the information from the previously rejected samples. While this minimizes the number of evaluations of π, the acceptance probabilities update in a manner depending on the target π, making the runtime difficult to analyze. Alternatively, the upper bound can be updated in a deterministic manner such as in Leydold et al. (2002), which makes understanding the runtime much simpler. While deterministic updates require more evaluations of π, they can potentially result in upper bounds that converge to π much faster, resulting in a tradeoff.

With adaptive rejection sampling, the marginal probability of accepting a sample at iteration n is $p_n = \frac{1}{c_n} \int \pi(x)\ dx$. However, as the acceptance probability changes over time, the runtime T to accept one sample is no longer geometric, but has pmf $P(T = k) = p_t \prod_{i=1}^{k-1}(1 - p_i)$, for $k = 1, 2, 3, \ldots$.

3. Privacy risk of rejection sampling

In this section, we characterize the privacy cost of a rejection sampler when we allow the adversary to have access to both the accepted sample as well as the runtime. Recall that if a rejection sampler is run until acceptance, then the accepted sample is an exact sample from the target distribution. Thus, the only increased privacy risk from using this algorithm is due to the runtime. We will measure the privacy risk of this side-channel in terms of ϵ-DP, (ϵ, δ)-DP, and f-DP. We show that for the exponential mechanism, the privacy cost of a rejection sampler's runtime is non-negligible.

Assumption 6 *For a rejection sampler, we assume that along with the published accepted sample, the runtime is also available to an attacker. We assume that for all databases D 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. Thus for the rest of the paper, the runtime will simply refer to the number of iterations in the sampler.*

Note that while the proposal distribution $U_D(x)$, target $\exp(g_D(x))$, and threshold c_D may all depend on D, none are directly available to the attacker.

Remark 7 *Many utility functions used in the exponential mechanism can be expressed as empirical risks (Bassily et al., 2014a,b; Reimherr and Awan, 2019; Wang et al., 2019). In this case, assuming that the database size n is fixed, ensuring that the time to evaluate $g_D(x)$ is constant is equivalent to ensuring that the contributions to the empirical risk from each individual take constant time. This is in line with the techniques used in Haeberlen et al. (2011) who split each query into sub-queries which are evaluated on each member of the dataset.*

First we will study the privacy cost of the rejection sampling runtime in terms of ϵ-DP. Proposition 9 states that the runtime of a rejection sampler violates ϵ-DP unless the probability of acceptance is constant across databases. To prove this, recall that ϵ-DP is measured by the max-divergence. Lemma 8 shows that the symmetric max-divergence between two geometric random variables is unbounded whenever the parameters differ, and the proposition follows easily from this.

Lemma 8 *Let $p, q \in (0,1)$ and let $X \sim \mathrm{Geom}(p)$ and $Y \sim \mathrm{Geom}(q)$. Then*

$$D_\infty(X||Y) = \begin{cases} \log(p/q) & \text{if } p \geq q \\ \infty & \text{if } p < q. \end{cases}$$

Thus, $D_\infty^S(X, Y) = \infty$ whenever $p \neq q$.

Proof As all geometric random variables, with parameter in $(0,1)$, are equivalent measures on the positive integers, it suffices to determine an upper bound on $\log \frac{P(X=k)}{P(Y=k)}$ for $k \in \{1, 2, \ldots\}$. This quantity can be expressed as

$$\log \frac{P(X=k)}{P(Y=k)} = \log \frac{(1-p)^{k-1}p}{(1-q)^{k-1}q} = \log\left(\frac{p(1-q)}{q(1-p)}\right) + k \log\left(\frac{1-p}{1-q}\right).$$

We see that this quantity is linear in k. The slope is non-positive if and only if $p \geq q$, in which case the maximum value is achieved at $k = 1$, giving the value $\log(p/q)$. When $p < q$, the slope is positive, and as $k \to \infty$, the quantity is unbounded. ■

Proposition 9 *Let $\{M_D \mid D \in \mathscr{D}\}$ be a privacy mechanism, let p_D be the probability of acceptance for a rejection sampler run on M_D, call T_D the runtime of the rejection sampler which is distributed $\mathrm{Geom}(p_D)$, and call X the accepted sample. If there exists $D, D' \in \mathscr{D}$ such that $d(D, D') \leq 1$ and $p_D \neq p_{D'}$, then the mechanism that releases (X, T) does not satisfy ϵ-DP for any $\epsilon > 0$.*

Proof By post-processing (Proposition 5), we get a lower bound on the privacy cost by only considering T. If there exists D and D' such that $d(D, D') \leq 1$ and $p_D \leq p_{D'}$, then by Lemma 8 the symmetric max-divergence is unbounded, and the result follows. ■

Theorem 10 gives a more precise characterization of the privacy loss due to rejection sampling as measured by f-DP and (ϵ, δ)-DP. For the former, we bound the tradeoff function

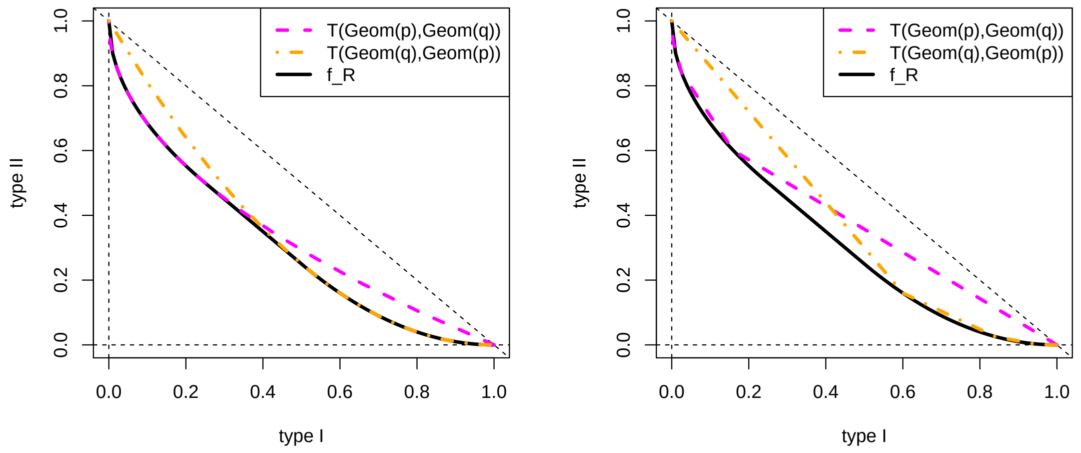


Figure 2: The tradeoff functions of $T(\text{Geom}(p), \text{Geom}(q))$ and $T(\text{Geom}(q), \text{Geom}(p))$, along with f_R from Theorem 10. We fix $R = 2$. In the left plot $q = .1$ and in the right $q = .6$. $p = 1 - (1-q)^R > q$. We see that the approximation f_R is more accurate for smaller q.

of the geometric variables with the tradeoff function for exponential variables, which allows for a simpler formula. This bound is tighter for small acceptance probabilities. We use the likelihood ratio property of the exponential distribution along with some properties of convex functions to get the formula in equation (1). We then use Proposition 5 to convert the f-DP guarantee to (ϵ, δ)-DP guarantees.

Theorem 10 *Let $(\mathscr{D}, d)$ 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 $T_D \sim$ 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*

1. *satisfies f_R-DP, where*

$$f_R(\alpha) = \begin{cases} 1 - \alpha^{1/R} & \alpha \leq R^{R/(1-R)} \\ -\alpha + R^{R/(1-R)} + 1 - R^{1/(1-R)} & R^{R/(1-R)} < \alpha < 1 - R^{1/(1-R)} \\ (1-\alpha)^R & \alpha \geq 1 - R^{1/(1-R)}, \end{cases} \tag{1}$$

2. *satisfies $(\epsilon, \delta(\epsilon))$-DP for all $\epsilon \geq 0$, where $\delta(\epsilon) = (1 - 1/R)\exp\left(\frac{-\epsilon - \log(R)}{R-1}\right)$,*

3. *satisfies $(\epsilon(\delta), \delta)$-DP for all $0 < \delta \leq (R-1)R^{R/(1-R)}$, where $\epsilon(\delta) = \log(1/R) + (R-1)(\log(1/\delta) + \log(1 - 1/R))$.*

Proof We begin by establishing the form of f_R, and then use Proposition 4 to produce (ϵ, δ)-DP guarantees. We first show that by bounding the tradeoff function of the exponential distribution, we get bounds for geometric variables as well. Call $\lambda_D = -\log(1-p_D)$. Recall that if $X_D \sim \text{Exp}(\lambda_D)$, then $\lfloor X_D \rfloor + 1 \sim \text{Geom}(p_D)$. By Proposition 5, we have that

$T(\mathrm{Exp}(\lambda_D), \mathrm{Exp}(\lambda_{D'})) = T(X_D, X_{D'}) \leq T(\lfloor X_D \rfloor + 1, \lfloor X_{D'} \rfloor + 1) = T(\mathrm{Geom}(p_D), \mathrm{Geom}(p_{D'}))$, where $T(\cdot, \cdot)$ represents the tradeoff function.

Next, we will derive the tradeoff function $T(\mathrm{Exp}(\lambda_D), \mathrm{Exp}(\lambda_{D'}))$ assuming that $\lambda_D > \lambda_{D'}$. Let $p_{\lambda_D}(x)$ be the pdf of $\mathrm{Exp}(\lambda_D)$. Note that $(p_{\lambda_{D'}}(x)/p_{\lambda_D}(x)) = (\lambda_{D'}/\lambda_D)\exp(x(\lambda_D - \lambda_{D'}))$ is an increasing function of x. By the Neyman Pearson Lemma, the most powerful test has a rejection region of the form $x \geq \tau$. The type I error is $\alpha = \exp(-\lambda_D \tau)$ and type II is $\beta = 1 - \exp(-\lambda_{D'}\tau)$. Expressing β as a function of α gives $\beta = 1 - \alpha^{\lambda_{D'}/\lambda_D} \geq 1 - \alpha^{1/R}$. Thus, we have that $T(X_D, X_{D'}) \geq 1 - \alpha^{1/R}$. We also need a lower bound on $T(X_{D'}, X_D)$. Note that $T(X_{D'}, X_D)$ is the inverse of $T(X_D, X_{D'})$. By taking the inverse of $1 - \alpha^{1/R}$, we have $T(X_{D'}, X_D) \geq (1-\alpha)^R$.

To get a single bound on both $T(X_D, X_{D'})$ and $T(X_{D'}, X_D)$, we take the convex hull of $\min\{1 - \alpha^{1/R}, (1-\alpha)^R\}$, which we claim has the form $f_R(\alpha)$ as stated in 1. To this end, we first verify that

$$1 - \alpha^{1/R} \leq 1 - R\alpha \leq (1-\alpha)^R, \tag{2}$$

for all $0 \leq \alpha \leq R^{R/(1-R)}$, so that over this range of α, the convex hull just equals $1 - \alpha^{1/R}$ as required by the first line of Equation (1). To establish the first inequality of Equation (2), note that $f(\alpha) = 1 - \alpha^{1/R}$ is a convex function; this can be seen either by differentiating it twice, or from the fact that it is a tradeoff function. The straight line $1 - R\alpha$ intersects this curve at $\alpha = 0$ and $\alpha = R^{R/1-R}$, and for intermediate values of α, forms a chord segment. From convexity, this chord lies above the curve. For the second inequality of Equation (2), observe that $(1-\alpha)^R$ is also convex. We can easily verify that the line $1 - R\alpha$ is the tangent at $\alpha = 0$. The second inequality then follows from the fact that a convex function is lower bounded by its tangent. This justifies the first line of $f_R(\alpha)$ in Equation (1). By symmetry, we also have that the third line is correct.

For the middle inequality, we note that the curves $1 - \alpha^{1/R}$ and $(1-\alpha)^R$ have slope -1 at the points $R^{R/(1-R)}$ and $1 - R^{1/(1-R)}$ respectively. It is easily verified that the straight line $g(\alpha) = -\alpha + R^{R/(1-R)} + 1 - R^{1/(1-R)}$ intersects the two curves at these two points, and has slope -1. It is thus tangent to both curves, and from convexity, lies below both of them. Altogether, we conclude that $f_R(\alpha)$ is the appropriate convex hull.

To get the formulas in 2. and 3., recall that the mechanism satisfies (ϵ, δ)-DP if the line $(1-\delta) - \exp(\epsilon)\alpha$ is a lower bound for the tradeoff function $f_R(\alpha)$. To get the tightest (ϵ, δ)-DP guarantees, we characterize the supporting linear functions. By symmetry, it suffices to determine the tangent lines of $1 - \alpha^{1/R}$ for values $0 \leq \alpha \leq R^{R/(1-R)}$. We calculate the derivative as $\frac{d}{d\alpha}(1 - \alpha^{1/R}) = \frac{-1}{R}\alpha^{1/R-1}$. Set $-\exp(\epsilon) = \frac{-1}{R}\alpha^{1/R-1}$, which has the solution $\epsilon = \log(1/R) + (1/R - 1)\log\alpha$.

The line with slope $-\exp(\epsilon) = \frac{-1}{R}\alpha^{1/R-1}$ that passes through $(\alpha, 1 - \alpha^{1/R})$ is $y - (1 - \alpha^{1/R}) = \frac{-1}{R}\alpha^{1/R-1}(x - \alpha)$, which has y-intercept $1 - \delta = 1 - \alpha^{1/R}(1 - 1/R)$, giving $\delta = \alpha^{1/R}(1 - 1/R)$. Eliminating α from the equations $\epsilon = \log(1/R) + (1/R - 1)\log\alpha$ and $\delta = \alpha^{1/R}(1 - 1/R)$ gives the expressions in parts 2. and 3. in the theorem statement. Note that $\delta(0) = (R-1)R^{R/(1-R)}$, so for any $\delta > \delta(0)$, the mechanism satisfies $(0, \delta)$-DP, but this is a strictly weaker guarantee than $(0, \delta(0))$-DP. ■

The approximation in Theorem 10 improves for smaller probabilities of acceptance, as seen in Figure 2. Intuitively, this is because the approximation of a geometric variable as an

$\delta =$	.1	.01	.001	10^{-4}	10^{-5}	10^{-6}
$R = 2$	.916	3.22	5.52	7.82	10.13	12.43
$R = 1.1$	0	.125	.356	.59	.82	1.05

Table 1: The $(\epsilon(\delta), \delta)$-DP guarantee for a simple rejection sampler, where R (defined in Theorem 10) is either 2 or 1.1. The values $\epsilon(\delta)$ appear in the table for each combination of δ and R.

exponential is more accurate for smaller probabilities of acceptance. As rejection samplers typically have small rejection probabilities, the privacy guarantees of Theorem 10 are quite accurate for rejection samplers of interest. In Table 1, we give a few examples converting the quantity R to (ϵ, δ)-DP guarantees. We see that even with a small R of 1.1, there is a nontrivial privacy cost. In Example 13, we explore what values of R we may expect in practice.

Corollary 11 *Let $(\mathscr{D}, d)$ be a metric space of databases, M_D a privacy mechanism which satisfies f-DP and T_D the runtime of a rejection sampler for M_D which has acceptance probability p_D. Call $R = \sup_{d(D,D')\leq 1} \frac{\log(1-p_D)}{\log(1-p_{D'})}$. Then the privacy cost of M_D along with the runtime is $f_R \otimes f$, where f_R is defined in Theorem 10 and $\otimes$ is the* tensor product *of two tradeoff functions (Dong et al., 2022, Definition 5).*

In Corollary 11, the tensor product $f \otimes g$, where $f = T(P, Q)$ and $g = T(P', Q')$ is defined as $f \otimes g = T(P \times P', Q \times Q')$, where $P \times P'$ is the product distribution (Dong et al., 2022, Definition 5). In general, it is challenging to derive a closed form of $f \otimes g$.

Remark 12 (Rejection sampling is trivial for location-scale) *For some distributions, it is easy to build a rejection sampler, with constant acceptance probability. For example, suppose that the mechanism $\{M_D \mid D \in \mathscr{D}\}$ is location-scale (e.g., K-norm mechanisms: Hardt and Talwar, 2010; Awan and Slavković, 2020). In this case, we build a rejection sampler for a default distribution in the family, and transform after sampling. Then we have a rejection sampler where the acceptance rate is independent of the dataset.*

While Theorem 10 describes the privacy cost of a rejection sampler's runtime, it is phrased in terms of the quantity R, which may be unintuitive. In the following example, we show that for a generic exponential mechanism, with an arbitrary set of proposal distributions, R is lower bounded by $\exp(\epsilon)$, and may even be infinite.

Example 13 (Exponential mechanism) *As shown in McSherry and Talwar (2007), the exponential mechanism results in a target distribution of the form $\widetilde{\pi}_D = \exp(g_D(x))$, which usually satisfies $\exp(-\epsilon/2) \leq \frac{\widetilde{\pi}_{D'}(x)}{\widetilde{\pi}_D(x)} \leq \exp(\epsilon/2)$ for adjacent databases D and D' (the integrating constants may also differ by a factor of at most $\exp(\pm\epsilon/2)$). Let $\mathscr{U}$ be a family of proposal distributions, and for each database D, let c_D and U_D be the optimal proposal distribution from $\mathscr{U}$ such that $\widetilde{\pi}_D \leq c_D U_D(x)$, where by optimal, we mean that the acceptance probability $p_D = \frac{\int \widetilde{\pi}_D(x)\ dx}{c_D}$ is maximized; or equivalently c_D is minimized. Then, from the* *following inequality,*

$$\widetilde{\pi}_{D'}(x) \leq \exp(\epsilon/2)\widetilde{\pi}_D(x) \leq \exp(\epsilon/2)c_D U_D(x),$$

we see that $\exp(\epsilon/2)c_D$ *and* U_D *offer a (potentially inferior) proposal distribution for* $\widetilde{\pi}_{D'}$*. Using this relationship between the proposal distributions of* D *and* D'*, we can give a bound for the acceptance probability for* D' *based on the acceptance probability for* D*:*

$$p_{D'} = \frac{\int \widetilde{\pi}_{D'}(x)\ dx}{c_{D'}} \geq \frac{\int \widetilde{\pi}_{D'}\ dx}{\exp(\epsilon/2)c_D} \geq \frac{\exp(-\epsilon/2)\int \widetilde{\pi}_D\ dx}{\exp(\epsilon/2)c_D} = \exp(-\epsilon)p_D.$$

Call p^* *the highest acceptance probability over all possible databases* D*. Then the quantity* R *that appears in Theorem 10 can be expressed as*

$$R = \frac{\log(1-p^*)}{\log(1-\exp(-\epsilon)p^*)}. \tag{3}$$

Note that as $p^* \to 1$ *in Equation* (3)*,* R *diverges to infinity. We can also get a lower bound on* R*:*

$$R = \frac{\log(1-p^*)}{\log(1-\exp(-\epsilon)p^*)} \geq \lim_{p\to 0}\frac{\log(1-p)}{\log(1-\exp(-\epsilon)p)} \overset{L'H}{=} \lim_{p\to 0}\frac{1-\exp(-\epsilon)p}{\exp(-\epsilon)(1-p)} = \exp(\epsilon), \tag{4}$$

where $\overset{L'H}{=}$ *indicates the use of L'Hôpital's rule, and we used the fact that* $\log(1-p)/\log(1-\exp(-\epsilon)p)$ *is increasing in* p *for all* $p \in (0,1)$ *and* $\epsilon > 0$*; to see this, we compute the derivative with respect to* p*:*

$$\frac{(1-p)\log(1-p) - (\exp(\epsilon)-p)\log(1-p\exp(-\epsilon))}{(\exp(\epsilon-p)(1-p)(\log(1-p\exp(-\epsilon)))^2}. \tag{5}$$

We see in (5) *that the denominator is positive so long as* $0 < p < 1$*. The numerator of* (5) *can be expressed as*

$$\sum_{n=2}^{\infty} p^n\left(\frac{1}{n(n-1)}\right)(1-\exp(-\epsilon(n-1))),$$

which we can see is positive and finite for all $\epsilon > 0$ *and* $p \in (0,1)$*.*

Remark 14 (Parallelization and batching) *Suppose that we have a simple rejection sampler targeting* $\widetilde{\pi}_D$ *with acceptance probability* p_D*. We could consider a parallelized implementation as follows: run the sampler on* k *nodes; when the first sample is accepted, return the sample and the runtime and abort the other instances of the sampler. In this scheme, the runtime is distributed as* $\min\{G_1,\dots,G_k\} \sim \mathrm{Geom}(1-(1-p_D)^k)$*, where* G_i *are independent* $\mathrm{Geom}(p_D)$ *random variables. Now, suppose for two adjacent databases* D *and* D' *that*

$$\frac{\log(1-p_D)}{\log(1-p_{D'})} = R,$$

which is the quantity in Theorem 10 that governs the privacy cost of the runtime. Then, in the parallelized scheme, we have

$$\frac{\log(1-[1-(1-p_D)^k])}{\log(1-[1-(1-p_{D'})^k])} = \frac{\log((1-p_D)^k)}{\log((1-p_{D'})^k)} = \frac{k\log(1-p_D)}{k\log(1-p_{D'})} = R.$$

We see that parallelization does not affect the privacy cost of the runtime.

Similarly, one could decide to run the rejection sampler for a fixed number of iterations, say m *before checking if one of the samples is accepted, and then repeating if necessary. This may be useful in combination with parallelization, since communication between the nodes could be a bottleneck. With batching, the runtime until a sample is accepted is* $\mathrm{Geom}(1-(1-p_D)^m)$*, which is the same runtime as in the parallelization. By the same reasoning, batching also does not affect the privacy cost of the runtime.*

In Section 4, we develop samplers with data-independent runtime. As such, parallelizing or batching the samplers in the manner described above maintains the property that the runtime is data-independent, while potentially giving a significant speed up.

3.1 Privacy risk of adaptive rejection sampling

In this section, we analyze the privacy risk of adaptive rejection samplers. 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. (2002). We show that unless the acceptance probabilities converge in a strong sense, an adaptive rejection sampler will not satisfy ϵ-DP for any finite ϵ.

Proposition 15 *Let* $\mathscr{D}$ *be the space of databases and* $\{M_D \mid D \in \mathscr{D}\}$ *a privacy mechanism which satisfies* ϵ*-DP. Let* $(p_i^D)_{i=1}^\infty$ *be the weakly-increasing sequence of acceptance probabilities for an adaptive rejection sampler for* M_D*. Call* T_D *the runtime of the adaptive sampler for* M_D*, which has pmf* $P(T_D = t) = p_t^D \prod_{i=1}^{t-1}(1-p_i^D)$*. Then releasing a sample from* M_D *as well as the runtime* T_D *satisfies* $(\epsilon+\epsilon_T)$*-DP, where*

$$\epsilon_T \geq \log(p_t^D/p_t^{D'}) + \sum_{i=1}^{t-1} \log\left(\frac{1-p_i^D}{1-p_i^{D'}}\right),$$

for all $t \geq 1$ *and all* $d(D,D') \leq 1$*. If there exists a constant* c *such that* $p_1^D \geq c > 0$ *for all* D*, then the value* ϵ_T *is finite if and only if the sequence* $\left(\sum_{i=1}^t \log \frac{1-p_i^D}{1-p_i^{D'}}\right)_{t=1}^\infty$ *is universally bounded for all* $d(D,D') \leq 1$*.*

Proof For readability, we set $p_i := p_i^D$ and $q_i := p_i^{D'}$. We require that $\log \frac{P(T_D=t)}{P(T_{D'}=t)} \leq \epsilon_T$ for all $d(D,D') \leq 1$ and all $t = 1, 2, \ldots$. A little algebra gives the expression for ϵ_T.

Next, ϵ_T is finite if and only if $\log \frac{P(T_D=t)}{P(T_{D'}=t)}$ is bounded above and below for all $d(D,D') \leq 1$. Equivalently, this requires $\log \frac{p_t \prod_{i=1}^{t-1}(1-p_i)}{q_t \prod_{i=1}^{t-1}(1-q_i)} = \log\left(\frac{p_t}{q_t}\prod_{i=1}^{t-1}\left(\frac{1-p_i}{1-q_i}\right)\right)$ be universally bounded above and below for all t. Since $\frac{p_t}{q_t}$ is bounded below by c and above by $1/c$, the previous quantity is bounded if and only if $\log\left(\prod_{i=1}^{t-1}\left(\frac{1-p_i}{1-q_i}\right)\right) = \sum_{i=1}^{t-1}\log\left(\frac{1-p_i}{1-q_i}\right)$ is bounded for all t. Relabelling $t-1$ to t gives the final result. ■

Proposition 15 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. In the following example, we explore a few special cases to highlight when we can or cannot expect the condition in Proposition 15 to hold.

Example 16 • *If there exists i such that $p_i = 1$ whereas $q_i < 1$ or vice versa, then $\epsilon_T = \infty$.*

- *Suppose that $(1-q_i) = \alpha(1-p_i)$ where $\alpha \in (0,1)$. Then the above series is $\sum_{i=1}^t \log \frac{1-p_i}{1-q_i} = \sum_{i=1}^t \log \alpha \to \infty$.*
- *To see that it is not sufficient for $\lim_{i\to\infty} \frac{1-p_i}{1-q_i} = 1$, consider the following: let $(1-p_i)$ be any decreasing sequence with values in $(0,1)$. Set $(1-q_i) = \exp(-1/i)(1-p_i)$. Then $\log\left(\frac{1-p_i}{1-q_i}\right) = 1/i$ and so $\frac{1-p_i}{1-q_i} \to 1$. However, the sequence of partial sums $\sum_{i=1}^t \log\left(\frac{1-p_i}{1-q_i}\right) = \sum_{i=1}^t 1/i$ diverges, and so the max-divergence is infinite.*

Remark 17 *In Proposition 15, convergence of the series $\sum_{i=1}^\infty \log \frac{1-p_i}{1-q_i}$ is sufficient but not necessary. It is possible that the sequence of partial sums is bounded but does not converge.*

Note that for most adaptive rejection samplers, it is difficult to derive expressions for p_i, so it may not even be possible to verify whether the condition in Proposition 15 holds or not. The takeaway is that in general, an adaptive rejection sampler is not guaranteed to preserve privacy unless it is carefully designed to do so.

4. Rejection samplers with data-independent runtime

The previous section showed that a rejection sampler (either simple or adaptive) can result in an arbitrary amount of privacy loss through the runtime. The most direct way to avoid this is to ensure that the runtime does not depend on the dataset. Haeberlen et al. (2011) propose making the runtime a constant, though this is not strictly necessary. Rather, when the runtime is a random variable (as with rejection sampling), we simply need that its distribution does not depend on the dataset.

In this section we propose three modifications of the rejection sampling algorithm to ensure data-independent runtime. The first method, which requires the weakest assumptions, fixes the number of iterations independent of the dataset, based on a worst-case acceptance probability. This method has a constant runtime, but there is a small probability that a sample is not accepted, and we quantify the additional privacy cost. The second method is based on the memoryless property of the geometric distribution, and introduces an additive random wait-time. This approach however requires the integrating constant of the target distribution corresponding to the current database, as well as the acceptance probability of a worst-case database, which is often not realistic. The third method avoids this by using instead upper and lower bounds for the target densities of all databases, chosen so that the ratio of the area for the upper and lower bounds is constant across databases. Finally, we propose an adaptive rejection sampler with data-independent runtime, which is a modification of the (nearly) minimax optimal sampler of Achddou et al. (2019). Our adaptive sampler is entirely automated, and only requires that the family of target densities is log-Hölder with fixed and known parameters

We show in Section 5 that many commonly studied privacy mechanisms satisfy the assumptions of our methods allowing for our privacy-preserving rejection samplers to be applied.

4.1 Constant runtime, truncated rejection sampling

One clear way to remove the privacy leak due to the runtime is to choose a number of iterations independent of the database, based on a worst-case estimate of the acceptance probability across all databases. We then run the sampler for that many iterations, and publish one of the accepted samples. In this case, the runtime is fixed, and does not leak any privacy. However, it is not guaranteed that an accepted sample is found within the pre-determined number of iterations, and the probability of this event *does* depend on the database. This probability can be reduced by increasing the number of iterations, but this also increases the runtime of the algorithm.

Of the methods we propose, the algorithm in Proposition 18 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 18 characterizes the increased cost to privacy of the truncated sampler in terms of (ϵ, δ)-DP.

Proposition 18 *Let $\{M_D \mid D \in \mathscr{D}\}$ be a family of privacy mechanisms satisfying (ϵ_0, δ_0)-DP and (U_D, c_D) be such that $\widetilde{\pi}_D \leq c_D U_D$ where $\widetilde{\pi}_D$ is an unnormalized density for M_D. Assume that $\alpha_0 \leq 1/c_D \int \widetilde{\pi}_D(x)\, dx$ 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, publish the first one; if not, publish 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.*

Proof First note that the runtime is constant for all D, so there is no privacy leak there. Next, note that conditional on the event that an accepted proposal is found, there is no additional privacy leak. So, we need to determine the probability that an accepted proposal is not found:

$$P(\text{none accepted}) = (1 - P(\text{accept}))^N \leq (1 - \alpha_0)^N = (1 - \alpha_0)^{\frac{\log(\delta)}{\log(1-\alpha_0)}} = \delta.$$

By itself, simply publishing whether a sample is accepted or not satisfies $(0, \delta)$-DP. By post-processing (Proposition 5), we can upper bound the privacy cost by instead considering if we observe both an output from M_D as well as whether the algorithm has accepted or rejected a sample. This is a composition of an (ϵ_0, δ_0)-DP mechanism with a $(0, \delta)$-DP mechanism. By composition (Dwork et al., 2014, Theorem 3.16) the result satisfies $(\epsilon_0, \delta_0 + \delta)$-DP. ■

A benefit of the algorithm in Proposition 18 is that it can be vectorized and is easily parallelized. Another benefit is that N grows only in the log of $1/\delta$. By increasing the number of iterations N, the increased δ can be reduced exponentially. 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 reduces both the privacy and utility. If no samples are accepted, then the output does not follow the correct distribution, introducing error in the sampling approximation. We see that we are able to remove the runtime side-channel, but at the cost of a small "delta" and loss of utility. In the next two subsections, we show that with slightly stronger assumptions, we are able to obtain both perfect sampling as well as data-independent runtime.

4.2 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 (or a lower bound on the acceptance probability).

The benefit of this method over the truncated rejection sampler is that a sample from the correct distribution is guaranteed, and the runtime is still independent of the database. The downside is that the acceptance probability (or equivalently the integrating constant) for the present database is required as well as a bound on the worst-case acceptance probability. Typically, rejection samplers do not assume that the integrating constant is known, however for low dimensional problems (e.g., ≤ 3), it may be possible to numerically evaluate the integral.

Lemma 19 illustrates the memoryless property of the geometric distribution. Given a simple rejection sampler with acceptance probability q, we can add a random wait time to result in a total runtime that is distributed as $\mathrm{Geom}(p)$ for $p \leq q$. So, across databases, we can make all of the runtimes equal in distribution, calibrated to a worst-case acceptance probability.

Lemma 19 *Let $0 < p \leq q < 1$. Given $X_2 \sim \mathrm{Geom}(q)$, set $X_1 = X_2$ with probability p/q and otherwise $X_1 = X_2 + \Delta$, where $\Delta \sim \mathrm{Geom}(p)$. Then $X_1 \sim \mathrm{Geom}(p)$.*

Proof Let $t \in \{1, 2, \dots, \infty\}$. Then

$$\begin{aligned} P(X_1 = t) &= \frac{p}{q} P(X_2 = t) + (1 - p/q) P(X_2 + \Delta = t) \\ &= \frac{p}{q}(1-q)^{t-1} q + \frac{q-p}{q} \sum_{x=1}^{t-1} P(X_2 = x) P(\Delta = t - x) \\ &= p(1-q)^{t-1} + \frac{q-p}{q} \sum_{x=1}^{t-1} (1-q)^{x-1} q (1-p)^{t-x-1} p \\ &= p(1-p)^{t-1}, \end{aligned}$$

which is the pmf of $\mathrm{Geom}(p)$, as desired. To achieve the last line in the equations, we used the partial sum formula for a geometric series, and simplified the result. ■

Theorem 20 *Let $D \in \mathscr{D}$ be a database and $\{\pi_D \mid D\}$ be the normalized target densities. Assume that for each π_D, we have normalized densities $U_D(x)$ as well as constants c_D such*

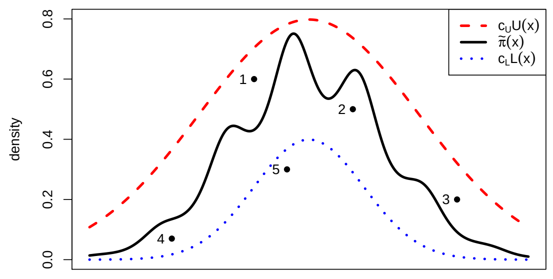


Figure 3: Example implementation of Algorithm 1. See Example 25 for details. The sample x_2 is accepted, but not published until x_5.

that for all x, $\pi_D(x) \leq c_D U_D(x)$. Suppose we know a constant c satisfying $c \geq \sup_D c_D$. 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 $Y \sim \mathrm{Unif}(0,1)$.*

3. *If $Y < c_D/c$, publish X, else wait for $\mathrm{Geom}(1/c)$ cycles before publishing X.*

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

As compared to the truncated rejection sampler of Section 4.1, Theorem 20 offers a perfect sampler with data independent runtime. This is ideal as there is no loss to either privacy or utility through either approximate samples or a runtime side-channel. However, the downside of this method is that the acceptance probability for the current database must be known. Assuming that the proposal is normalized, this is equivalent to knowing the integrating constant for the target . While this may not be too cumbersome for low-dimensional settings, it becomes computationally intractable for high-dimensional distributions. In the next section, we give an alternative set of assumptions to remove the requirement of the integrating constant.

Remark 21 *A similar alternative to Theorem 20 is as follows: during each step of the rejection sampler, if a sample is accepted, then with probability c_D/c report the sample, and with probability $1 - c_D/c$ do not report the sample. This results in the same runtime as Theorem 20.*

We remark that this alternative algorithm has a similar flavor to the randomized response mechanism, one of the oldest privacy mechanisms (Warner, 1965). While beyond the scope of this paper, it may be worth investigating whether there is any deeper connection between this privacy-aware rejection sampler and randomized response.

4.3 Implicit wait-time via squeeze function

In this section, we propose another method of producing an exact rejection sampler with data-independent runtime. Our method, described in Algorithm 1 and Theorem 22, avoids the need for the normalizing constant as in Theorem 20 by instead using a carefully tailored *squeeze* function. In the rejection sampling literature, a squeeze function is a lower bound on the target density which is assumed to be easy to evaluate, and which is used to speed up the computational time by avoiding evaluations of the target density when a proposed sample lies under the squeeze function (i.e. is rejected by the squeeze function, see Example 25). However, in this section, we will use the squeeze function not to speed up the computational time, but to slow it down; this will enable us to make the runtime equally distributed as in a worst-case setting.

For this method, we assume that for each unnormalized target density $\widetilde{\pi}_D$, we have normalized densities $U_D(x)$ and $L_D(x)$ as well as constants $c_{L,D}$ and $c_{U,D}$ such that for all x

$$c_{L,D} L_D(x) \leq \widetilde{\pi}_D(x) \leq c_{U,D} U_D(x),$$

and such that the ratio $c_{L,D}/c_{U,D}$ does not depend on D. Note that the latter condition is easy to enforce: if $c_{L,D}$ and $c_{U,D}$ are two valid constants, then so are $c^*_{L,D} < c_{L,D}$ and $c^*_{U,D} > c_{U,D}$. We then choose the value X_s that we publish based on the rejection sampler that targets $\widetilde{\pi}_D$ from U_D, but do not publish the sample until a value is accepted from L_D (i.e. the proposal lies under the squeeze function $c_{L,D}L_D$: see Example 25). Because of this modification, the runtime is determined only by the ratio c_L/c_U, which is assumed to be constant across databases. Thus, there is no additional privacy cost to using this sampler, since we get an exact sample with runtime independent of D. This method is similar to that of Section 4.2 in that there is an additive wait-time, but Algorithm 1 is able to do this implicitly, without knowing the acceptance probability for the current database. In Proposition 23, we show that the assumptions of Theorem 22 are strictly weaker than those of Proposition 20.

Algorithm 1 Privacy-aware rejection sampling via squeeze functions

```
INPUT: $\widetilde{\pi}$, $U$, $L$, $c_U$, and $c_L$ such that $c_L L(x) \leq \widetilde{\pi}(x) \leq c_U U(x)$ for all $x$
 1: Set anyAccepted=FALSE
 2: Sample $X \sim U(x)$
 3: Sample $Y \sim \text{Unif}(0,1)$
 4: if $Y \leq \frac{\widetilde{\pi}(X)}{c_U U(X)}$ and anyAccepted==FALSE then
 5:     Set $X_s = X$
 6:     Set anyAccepted=TRUE
 7: end if
 8: if $Y \leq \frac{c_L L(X)}{c_U U(X)}$ then
 9:     Publish $X_s$
10: else
11:     Go to 2.
12: end if
OUTPUT: $X_s$
```

Theorem 22 *Let $D \in \mathscr{D}$ be a database and $\{\widetilde{\pi}_D \mid D\}$ be the (unnormalized) target densities. Assume that for each $\widetilde{\pi}_D$, we have normalized densities $U_D(x)$ and $L_D(x)$ as well as constants $c_{L,D}$ and $c_{U,D}$ such that the ratio $c_{L,D}/c_{U,D}$ does not depend on D and such that* *for all x $c_{L,D}L_D(x) \leq \widetilde{\pi}_D(x) \leq c_{U,D}U_D(x)$. 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.*

Proof The published sample is determined by the condition $Y \leq \frac{\widetilde{\pi}_D(X)}{c_{U,D}U_D(X)}$, where $X \sim U_D(x)$ and $Y \sim \mathrm{Unif}(0,1)$. This is a simple rejection sampler, and so conditional on acceptance, $X \sim \pi_D$. However, a sample is not published until $Y \leq \frac{c_{L,D}L_D(X)}{c_{U,D}U_D(X)}$. This is a rejection sampler targeting $L_D(X)$, using the proposal $U_D(X)$ and threshold $c_{U,D}/c_{L,D}$. As such, the number of iterations is $\mathrm{Geom}(c_{L,D}/c_{U,D})$, which by assumption does not depend on D. ■

While the assumption of the squeeze functions in Theorem 22 may seem unintuitive, it is in fact strictly weaker than knowing the integrating constant for $\widetilde{\pi}_D$, as was required in Section 4.2, as shown in Proposition 23. In Section 4.4 and 5 we show that there are several natural instances of the exponential mechanism where the assumptions of Theorem 22 are satisfied.

Proposition 23 *Let $D \in \mathscr{D}$ be a database and $\{\pi_D \mid D\}$ be the normalized target densities. Assume that for each π_D, we have normalized densities $U_D(x)$ and constants $c_{U,D}$ such that $\pi_D(x) \leq c_{U,D}U_D(x)$. Choose a value $c \geq \sup_D c_{U,D}$. Then the squeeze function $L_D = \pi_D$, with constant $c_{L,D} = c_{U,D}/c$ satisfies the assumptions of Theorem 22, and the output of Algorithm 1 has distribution π_D and runtime* $\mathrm{Geom}(1/c)$.

Proof Since $c \geq c_{U,D}$, we have that $c_{L,D} = c_{U,D}/c \leq 1$. So, $c_{U,D}L_D(x) \leq \pi_D(x)$ for all x. Then, the runtime of Algorithm 1 is geometric with parameter $(c_{L,D}/c_{U,D}) = 1/c$, and the output of Algorithm 1 has the appropriate distribution as argued in the proof of Theorem 22. ■

In fact, the application of Algorithm 1 described in Proposition 23 is very similar to the variation of Theorem 20 described in Remark 21.

Remark 24 *Proposition 23 showed that the assumptions for the squeeze functions in Theorem 22 are actually strictly weaker than the assumptions needed in Section 4.2. Furthermore, it can be seen that the assumptions of Theorem 22 (assuming that we can evaluate the constant $c_{L,D}/c_{U,D}$) are strictly stronger than knowing the worst-case acceptance probability, which is needed for the truncated sampler of Section 4.1 – this is because the ratio $c_{L,D}/c_{U,D}$ is itself a lower bound on the worst-case acceptance probability.*

Example 25 *Figure 3 is an illustration of how Algorithm 1 works. We see an example of a target $\widetilde{\pi}$, which satisfies $c_L L(x) \leq \widetilde{\pi}(x) \leq c_U U(x)$ for constants c_L, c_U, a proposal function U and squeeze function L. The points (x_i, y_i) are sequentially drawn uniformly within the area under $c_U U$; equivalently, $x_i \sim U(x)$ and $y_i = u_i \cdot c_U U(x)$, where $u_i \overset{iid}{\sim}$* $\mathrm{Unif}(0,1)$. *Algorithm 1 processes these samples as follows: For the first pair, $y_1 > \widetilde{\pi}(x_1)$ so the sample is rejected. The second sample satisfies $y_2 \leq \widetilde{\pi}(x_2)$ so it is accepted (set $X_s = x_2$), but* *because $y_2 > c_L L(x_2)$ it is not published yet. The third sample is rejected since $y_3 > \widetilde{\pi}(x_3)$. The fourth sample satisfies $y_4 \leq \widetilde{\pi}(x_4)$, but since we already accepted x_2, we do not update X_s. Since $y_4 > c_L L(x_4)$ we still do not publish anything yet. Finally, $y_5 \leq c_L L(x_5)$ so we publish $X_s = x_2$.*

As noted in Theorem 22, the procedure results in $X_s \sim \pi$, but the runtime is distributed as $\mathrm{Geom}(c_L/c_U)$, which does not directly depend on $\widetilde{\pi}$.

4.4 Adaptive rejection sampler for log-Hölder densities

The previous three subsections proposed modifications to simple rejection samplers in order to remove the runtime side-channel. In this section, we use the squeeze method of Section 4.3 to develop an adaptive rejection sampler with data-independent runtime for log-Hölder densities. Our method, outlined in Algorithm 2, is entirely black box, requiring only Hölder parameters (s, H) that hold for every database, and is a modification of the (nearly) minimax optimal sampler of Achddou et al. (2019). 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)-Hölder for all datasets D: $|g_D(x) - g_D(y)| \leq H\|x - y\|^s$ for all D and for all $x, y \in C$. This setup differs from Achddou et al. (2019), who assume that the target itself is Hölder, rather than the log-target. This difference is important in order to derive upper and lower bounds that satisfy a property similar to Theorem 22. We point out in Remark 29 that the log-Hölder assumption, with the same s and H across all datasets, is natural for many privacy mechanisms, and many instances of the exponential mechanism in the literature satisfy this assumption.

At a high-level, given evaluations of $g_D(x)$ at a finite set of locations, the log-Hölder assumption allows us to construct piecewise-constant upper and lower bounds on $g_D(x)$ and therefore the target density. Importantly, these bounds can be constructed so that the ratio of their associated normalization constants is independent of the database D. Then, in the fashion of Algorithm 1, by proposing from the upper bound, and stopping only on accepting from the lower bound, we can have a database-independent runtime. Following each proposal, we add a new location to our set of evaluations of $g_D(x)$, tightening the lower and upper bounds, and ensuring the acceptance probability increases each iteration. We describe these steps in detail in Algorithm 2.

Theorem 26 *Let $\mathscr{D}$ be a space of databases and $\{\widetilde{\pi}_D = \exp(g_D) \mid D\}$ be the unnormalized target densities, which have support on a bounded convex set C. Suppose that for all D, g_D is (s, H)-Hölder with norm $\|\cdot\|$ on C. Then Algorithm 2 results in N i.i.d. samples from $\widetilde{\pi}$ and has runtime between published samples which does not depend on D. If the mapping P_T and the update procedure to generate Z are chosen in a way that $\sup_{x \in C}\|x - P_T(x)\| \to 0$, then the probability of publishing an accepted sample in a given iteration converges to* 1.

Proof The quantity $\hat{r}$ is an upper bound on the maximum difference between $g_D(x)$ and $\hat{g}(x)$, by the Hölder assumption. So, at any point in the algorithm, since g_D is Hölder, and by the definition of $\hat{r}$, we have that

$$\exp(\hat{g}(x) - \hat{r}) \leq \exp(g_D(x)) \leq \exp(\hat{g}(x) + \hat{r}). \tag{6}$$

Using the notation of Theorem 22, we have that $c_{L,D} = \exp(-\hat{r})$, $L_D(x) = k\exp(\hat{g}(x))$, $c_{U,D} = \exp(\hat{r})$ and $U_D(x) = k\exp(\hat{g}(x))$, where $k = (\int_C \exp(\hat{g}(x))\, dx)^{-1}$. Since $c_{L,D}/c_{U,D} =$

Algorithm 2 Privacy-aware adaptive rejection

```
INPUT: $g$ an $(s,H)$-Hölder function on a bounded convex set $C \subset \mathbb{R}^d$ for some norm $\|\cdot\|$, initial evaluation points $\{(x_1, g(x_1)), \ldots, (x_n, g(x_n))\}$, 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 anyAccepted=FALSE, numSamples=0, and publishedSamples= $\emptyset$
 2: Set $S = \{(x_1, g(x_1)), \ldots, (x_n, g(x_n))\}$, and $T = \{x \mid (x, y) \in S \text{ for some } y\}$
 3: while numSamples< $N$ do
 4:     Define $\hat{g}(x) = g(P_T(x))$ for all $x \in C$ (note that this only requires evaluations of $g$ from $S$)
 5:     Set $\hat{r} \geq \sup_{x \in C} H\|x - P_T(x)\|^s$
 6:     Sample $X \sim \exp(\hat{g}(x))/(\int_C \exp(\hat{g}(x))\,dx)$
 7:     Sample $Y \sim \text{Unif}(0,1)$
 8:     if $Y \leq \exp(g(X))/\exp(\hat{g}(X) + \hat{r})$ and anyAccepted=FALSE then
 9:         Set $X_s = X$
10:         Set anyAccepted=TRUE
11:     end if
12:     if $Y \leq \exp(-2\hat{r})$ then
13:         Publish $X_s$ and append $X_s$ to publishedSamples
14:         Increment numSamples by 1
15:         Set anyAccepted=FALSE
16:     end if
17:     Choose $Z \in C \setminus T$ either randomly or deterministically based on only $T$, $H$ and $s$
18:     Append $(Z, g(Z))$ to $S$
19:     Append $Z$ to $T$
20: end while
    OUTPUT: publishedSamples, which can be published in a stream
```

$\exp(-2\hat{r})$ does not depend on D, by Theorem 22 the published samples are drawn independently from π_D and the runtime does not depend on D.

The probability of publishing a sample is $\exp(-2\hat{r})$. So, as long as $\|x - P_T(x)\|$ decreases as more samples Z are appended to T, we have $\hat{r} \to 0$ and thus the probability of publishing an accepted sample converges to 1. ■

Because the update step and the rejection step are separated, we can think about the best way to update the proposal function. Our goal should be to reduce $\hat{r}$ as quickly as possible. A simple, but naive solution would be to sample Z uniformly on C. Another approach would be to choose a sequence of $(x_i)_{i=1}^{\infty}$ such that for any N, the subset $(x_1)_{i=1}^{N}$ consists of approximately equally spaced points in C. This could be done intelligently using sequential space-filling experimental designs (e.g., Crombecq and Dhaene, 2010; Pronzato and Müller, 2012). For example, a greedy maximin solution would be to choose $z = \arg\sup_{z \in C} H\|z - P_T(z)\|^s$ (Pronzato and Müller, 2012), which maximizes the publishing probability for the next iteration. Computing the maximin solution may be possible in low-dimensions, but becomes expensive in high dimensional spaces.

As in Achddou et al. (2019), 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 Hölder 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. (2019, Definition 4). These modifications make the construction, evaluation, and sampling of the proposal $\exp(\hat{g})$ computationally efficient, even in high dimensions. The accept-reject steps (lines 6-16) and the update steps (lines 17-19) can be done in batches to avoid updating the function $\hat{g}$ too often, when it will not significantly improve the acceptance probability.

Remark 27 (Relative runtime) *We consider how the runtime of Algorithm 2 compares to a similar sampler without the privacy constraint. Recall that the acceptance probability of our sampler is* $\exp(-2\hat{r})$. *If we use the same proposal distribution, but base the acceptance criteria solely on the target, then the acceptance probability depends on the target. For a typical target density, we expect that the acceptance probability is approximately* $\exp(-\hat{r})$. *If this is the case, then as* $\hat{r} \to 0$, *the ratio of the rejection probabilities is*

$$\lim_{\hat{r}\to 0} \frac{1-\exp(-2\hat{r})}{1-\exp(-\hat{r})} = \lim_{\hat{r}\to 0} \frac{2\hat{r}}{\hat{r}} = 2,$$

where we use a series expansion of $\exp(-x)$ *about zero to evaluate the limit. This suggests that the cost of privacy is that the rejection probability is about double that of the non-private sampler.*

Another cost of the privacy-preserving adaptive sampler is the decoupling of the rejection and update steps. Roughly, we will need to evaluate g_D *twice as often—one for the update and one for the accept/reject step—as compared to non-private adaptive samplers, such as in Achddou et al. (2019). This additional cost is somewhat mitigated by the fact that the update points can be chosen in a more intelligent manner, potentially improving the rate of convergence of the proposal.*

Example 28 *We illustrate Algorithm 2 applied to the target* $\widetilde{\pi} = \exp(g(x))$, *where* g *is the 7-Lipschitz function* $g(x) = -3|x - 1/2| + (1/5)\sin(20x)$, *so* $H = 7$ *and* $s = 1$. *The update and sampling steps of Algorithm 2 are run in batches. First, 5 equally spaces points are used to approximate* $\exp(g(x))$ *by a piece-wise linear function* $\exp(\hat{g}(x))$. *The upper bound is* $\exp(\hat{g}(x) + \hat{r})$ *and the lower bound is* $\exp(\hat{g}(x) - \hat{r})$. *The top left plot of Figure 4, illustrates each of these functions. Next, five points* (x_i, y_i) *for* $i = 1, \ldots, 5$ *are sampled uniformly within the area under* $\exp(\hat{g}(x)+\hat{r})$, *as seen in the top right plot of Figure 4. The first value* y_1 *is below* $\exp(g(x_1))$, *but not below* $\exp(\hat{g}(x_1) - \hat{r})$, *so we set* $X_s = x_1$ *and set* `anyAccepted` $=$ *TRUE, but do not publish* X_s *yet. We reject* x_2. *Then as* $y_3 \le \exp(\hat{g}(x_3)-\hat{r})$, *we publish* $X_s = x_1$, *and set* `anyAccepted` $=$ *FALSE. We reject both* x_4 *and* x_5.

After this, we update the approximation $\hat{g}$ *using 15 equally spaced points. This grid is a superset of the 5-point grid, so we can reuse the previous evaluations. The new approximation and bounds are shown in the bottom left plot of Figure 4. Then* (x_i, y_i) *for* $i = 6, \ldots, 10$ *are sequentially sampled uniformly from the area under* $\exp(\hat{g}(x) + \hat{r})$, *illustrated in the bottom right plot of Figure 4. As* $\exp(\hat{g}(x_6) - \hat{r}) < y_6 \le \exp(g(x_6))$, *we set* $X_s = x_6$ *and* `anyAccepted` $=$ *TRUE, but do not publish* X_s. *Since* $y_7 \le \exp(\hat{g}(x_7)-\hat{r})$, *we publish* $X_s = x_6$ *at this time and set* `anyAccepted` $=$ *FALSE. Then since* $y_8 \le \exp(\hat{g}(x_8)-\hat{r})$, *we immediately publish* x_8. *We reject* x_9. *Last, as* $y_{10} \le \exp(\hat{g}(x_{10}) - \hat{r})$, *we also publish* x_{10}.

Note that by using equally spaced points, $\hat{r}$ *converges to zero rapidly, illustrating the benefit of using deterministically chosen points in the construction of* $\hat{g}$.

Remark 29 *There are several prior DP works on the exponential mechanism, where the utility function is assumed to be Lipschitz (a special case of Hölder), and where Algorithm 2 can be applied. Minami et al. (2016) assume Lipschitz and concave utility functions. Bassily et al. (2014a) and Bassily et al. (2014b) derive optimal DP mechanisms under the*

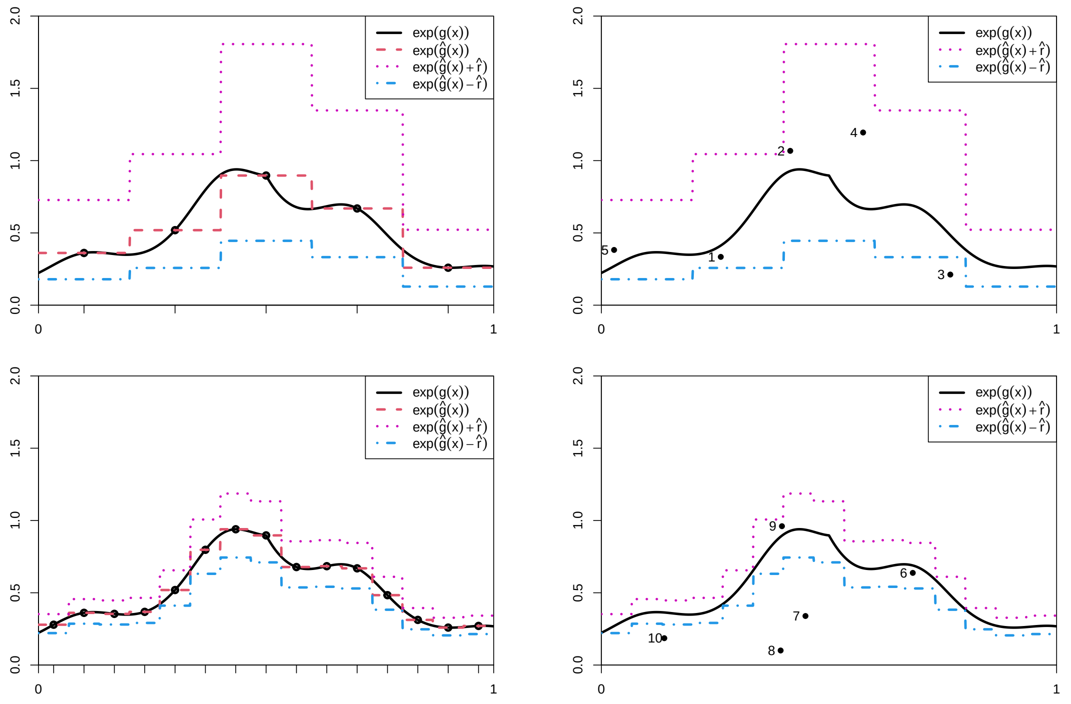


Figure 4: Implementation of Algorithm 2, as explained in Example 28. Plots progress in the normal reading order.

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 (2020) assume Lipschitz and L-smooth utility functions in the exponential mechanism.

5. Application to exponential mechanism sampling

In this section, we explore some instances of the exponential mechanism that satisfy the assumptions of the rejection samplers proposed in Section 4, allowing for a privacy-preserving implementation.

5.1 Strongly concave and L-smooth log-density

We first 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 (2020) assume. Both Awan et al. (2019) and Minami et al. (2016) assume strongly concave utility functions in the exponential mechanism. Other private empirical risk minimization works, while not working directly with the exponential mechanism, also commonly assume L-smooth and strong concavity (Kifer et al., 2012; Bassily et al., 2014a,b).

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 22.

Lemma 30 *Let $\widetilde{\pi}_D(x) \propto \exp(g_D(x))$ be the (unnormalized) target density, where $g_D : \mathbb{R}^d \to \mathbb{R}$ is twice-differentiable, α-strongly concave, and L-smooth. Call $x_D^* := \arg\max_x g_D(x)$. Using $\phi_d(x; \mu, \Sigma)$ to denote the pdf of $N_d(\mu, \Sigma)$, we have for all x,*

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

Furthermore, calling $c_{L,D} = \exp(g_D(x_D^*))\,(2\pi/L)^{d/2}$ and $c_{U,D} = \exp(g_D(x_D^*))\,(2\pi/\alpha)^{d/2}$, we have that $c_{L,D}/c_{U,D} = (\alpha/L)^{d/2}$, which does not depend on D.

Proof By strong concavity, we have that

$$-g_D(x) \geq -g_D(x^*) - \nabla g_D(x_D^*)^\top (x - x_D^*) + \frac{\alpha}{2}\|x_D^* - x\|_2^2 = g_D(x_D^*) + \frac{\alpha}{2}\|x_D^* - x\|_2^2,$$

since $\nabla g_D(x_D^*) = 0$. This implies that

$$\exp(g_D(x)) \leq \exp(g_D(x_D^*))\exp\left(-\frac{\|x_D^* - x\|_2^2}{2(1/\alpha)}\right).$$

Including the integrating constant for a multivariate normal distribution gives the upper bound.

Next, since g_D is L-smooth, we have that

$$\begin{aligned}-g_D(x) &= -g_D(x_D^*) - \nabla g_D(x_D^*)^\top (x - x_D^*) + \frac{1}{2}(x_D^* - x)^\top \nabla^2 g_D(\widetilde{x})(x_D^* - x)\\ &\leq -g_D(x_D^*) + \frac{L}{2}\|x_D^* - x\|_2^2,\end{aligned}$$

where $\widetilde{x}$ is between x_D^* and x, we used the fact that $\nabla g_D(x_D^*) = 0$, and that the eigenvalues of $\nabla^2 g_D$ are upper bounded by L. This implies that

$$\exp(g_D(x)) \geq \exp(g_D(x_D^*))\left(\frac{2\pi}{L}\right)^{d/2}\phi_d\left(x; x_D^*, L^{-1}I\right),$$

giving the lower bound. ■

Given the bounds in Lemma 30, we can now implement the squeeze-function rejection sampler of Section 4.3, since $c_{L,D}/c_{U,D}$ does not depend on D. As discussed in Proposition 23, generating these bounds is strictly easier than computing the integrating constant for the target, which is not needed in Lemma 30.

We could also implement the truncated sampler of Section 4.1, by using the bound $(\alpha/L)^{d/2}$ on the worst-case acceptance probability. However, since Theorem 22 is applicable, the truncated sampler is strictly worse as it incurs a price both in privacy as well as utility, whereas the squeeze sampler produces perfect samples.

There are many natural problem settings that fit the assumptions of Lemma 30, particularly in empirical risk minimization.

Example 31 (Strongly convex empirical risk minimization) *Suppose that the database can be written as a vector $D = (d_1, \ldots, d_n)$, where d_i is the contribution of individual i. Take as our utility function $g_D(x) = -\left(\sum_{i=1}^n \ell(x; d_i) + r(x)\right)$, where $\ell(x; d)$ is a twice-differentiable convex function which is L-smooth and satisfies $\sup_{d,d'} \sup_x |\ell(x;d) - \ell(x;d')| \leq \Delta$, and $r(x)$ is an α-strongly convex regularizer, which does not depend on the database D. For instance, we could take $r(x) = \frac{\alpha}{2}\|x\|_2^2$. Then the exponential mechanism samples from $\pi_D(x) \propto \exp(\frac{\epsilon}{2\Delta} g_D(x))$ and satisfies ϵ-DP.*

Note that g_D is nL-smooth and α-strongly concave for all D, so it fits the framework of Lemma 30. Such a setup is common in private empirical risk minimization (Kifer et al., 2012; Bassily et al., 2014a,b), and in particular for private regression problems (Kifer et al., 2012; Reimherr and Awan, 2019; Awan and Slavković, 2020).

5.2 K-norm gradient mechanism

An alternative to simply applying the exponential mechanism to a strongly concave utility function is the K-norm gradient mechanism (KNG), proposed in Reimherr and Awan (2019), also known as the gradient mechanism (Asi and Duchi, 2020a). KNG has been applied to applications such as geometric median estimation, and linear and quantile regression (Reimherr and Awan, 2019; Asi and Duchi, 2020a). Given an objective function $g_D(x)$, KNG samples from $\pi_D(x) \propto \exp(-\frac{\epsilon}{2\Delta}\|\nabla g_D(x)\|_K)$, where $\Delta \geq \sup_{d(D,D')\leq 1} \sup_x \|\nabla g_D(x) - \nabla g_{D'}(x)\|_K$, and where $\|\cdot\|_K$ is a chosen norm.

While the exponential mechanism with a strongly concave utility is naturally approximated by a Gaussian distribution (Awan et al., 2019), KNG is closely related to the K-norm distributions (Reimherr and Awan, 2019). The K-norm mechanism was introduced in Hardt and Talwar (2010), and were also studied in Awan and Slavković (2020).

Definition 32 (K-norm distribution: Hardt and Talwar, 2010) *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*

$$f(x; m, s) = c^{-1} \exp\left(-s^{-1}\|x - m\|_K\right),$$

where $c = (d!)s^d \mathrm{Vol}(K)$.

Under similar assumptions as those in Reimherr and Awan (2019, Theorem 3.1), Lemma 33 gives upper and lower bounds which satisfy the assumptions required for Theorem 22.

Lemma 33 *Let $\widetilde{\pi}_D(x) = \exp(-\|\nabla g_D(x)\|_2)$ be the unnormalized target density, where $g_D : \mathbb{R}^d \to \mathbb{R}$ is twice-differentiable, α-strongly convex, and L-smooth. Call $x_D^* := \arg\min_x g_D(x)$. Write $\psi_d(x; m, s)$ to denote the pdf of a d-dimensional K-norm distribution with location m, scale s, and ℓ_2 norm. Denote $\mathrm{Vol}_d(\ell_2) = \frac{2^d \Gamma^d(1+1/2)}{\Gamma(1+d/2)}$ the volume of the unit ℓ_2 ball in $\mathbb{R}^d$. Then for all x,*

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

Furthermore, calling $c_{L,D} = (d!)L^{-d}\mathrm{Vol}_d(\ell_2)$ and $c_{U,D} = (d!)\alpha^{-d}\mathrm{Vol}_d(\ell_2)$, we have that the ratio $c_{L,D}/c_{U,D} = (\alpha/L)^d$ does not depend on D.

Proof By strong convexity, we have that

$$\alpha\|x - x_D^*\|_2^2 \leq \langle \nabla g_D(x) - \nabla g_D(x_D^*), x - x_D^*\rangle = \langle \nabla g_D(x), x - x_D^*\rangle \leq \|\nabla g_D(x)\|_2 \cdot \|x - x_D^*\|_2,$$

where we used the fact that $\nabla g_D(x_D^*) = 0$ and Cauchy-Schwartz inequality. This implies that $\|\nabla g_D(x)\|_2 \geq \alpha\|x - x_D^*\|_2$, which gives the upper bound.

Next, as g_D is L-smooth, we have that

$$\|\nabla g_D(x)\|_2 = \|\nabla g_D(x) - \nabla g_D(x_D^*)\|_2 \leq L\|x - x_D^*\|_2,$$

which gives the lower bound. ■

Lemma 33 provides bounds that can be used to implement the sampler of Section 4.3. The acceptance probability when targeting the lower bound is $(\alpha/L)^d$, which is independent of D, as required. While we could implement the sampler of Section 4.1, as $(\alpha/L)^d$ is a bound on the worst-case acceptance probability, this method is strictly worse than the squeeze sampler, as discussed in Section 5.1. The empirical risk problems of Example 31 are also applicable to KNG, and offer several natural instances that satisfy the assumptions of Lemma 33.

Finally, note that for the KNG mechanism, if the underlying utility is L-smooth (not necessarily strongly concave), then the log-density is L-Lipschitz. As such, we can apply the adaptive rejection sampler of Section 4.4. If multiple i.i.d. samples are required, this can provide a very computationally efficient sampling method, while keeping the runtime data-independent.

6. Discussion

In this paper, we first characterized the privacy cost due to the runtime of both simple and adaptive rejection samplers in terms of ϵ-DP, (ϵ, δ)-DP, and f-DP. We found that the runtime of standard rejection samplers can result in an arbitrary increase in the privacy cost, motivating the need for privacy-aware samplers. We then proposed three novel modifications to simple rejection samplers with varying assumptions, which all resulted in data-independent runtime. We also developed a privacy-aware adaptive rejection sampler for log-Hölder densities.

There are three factors that influence the practicality of our algorithms, (1) *the scalability of rejection sampling:* Typically, the acceptance probability of a rejection samplers decays exponentially with data dimension, making them impractical for very high dimensional problems. However, imposing additional structure like log-concavity or log-Hölder on the target density, adaptive rejection samplers (like our proposed one) can be applicable to higher-dimensional problems. Such structural assumptions, as well as low- to moderate-dimensional problems are common in differential privacy applications. (2) *the additional cost of our differentially-private modifications of rejection sampling:* Our algorithms typically result in a reduction in the acceptance probability to match the worst-case dataset. This is unavoidable. However, our adaptive rejection sampler does not require any knowledge of this worst-case database. As such, the sample complexity of the runtime is the same as for a regular rejection sampler, but where the acceptance probability is the worst case. (3)

The additional book-keeping overhead in implementing our differentially private rejection samplers: All of our algorithms are minor modifications of existing simple or adaptive rejection samplers, and as such, this overhead is minimal.

Of our proposed modifications to the rejection sampler, the squeeze method of Section 4.3 is the most powerful. We showed in Section 5 that for many instances of the exponential mechanism, appropriate upper and lower bounds can be generated. Furthermore, our adaptive sampling scheme is also built on the squeeze sampler. In a way, Algorithm 1 can be viewed as a coupling of the sampler applied to the present database and a worst-case database. It is an open question whether similar couplings could be developed with even weaker assumptions.

An alternative to rejection sampling is *coupling from the past* (CFP) (Propp and Wilson, 1998), a modified MCMC approach. The benefit of CFP is that it is another perfect sampler, and could be a useful technique in designing privacy-aware samplers. The techniques used in this paper may be useful for determining the privacy cost of timing channel attacks on CFP and developing CFP algorithms with data-independent runtime. A variation on CFP is *perfect tempering*, which also results in perfect samplers (Møller and Nicholls, 1999; Daghofer and von der Linden, 2004; Brooks et al., 2006), and may be an another approach to developing privacy-aware samplers.

While in this paper we developed samplers whose runtime does not depend on the dataset, one could instead ask for the runtime to be differentially private by itself. Theorem 10 shows that a naive rejection sampler does have an inherent privacy cost, but one could also imagine altering the runtime to give a stronger privacy guarantee. A significant challenge with this approach is that we can only increase, but not reduce, the runtime. Due to this constraint, many existing DP techniques are not applicable. We leave it for future research to investigate mechanisms to privatize the runtime.

Acknowledgments

This work was supported in part by the National Science Foundation (NSF) grants SES 2150615, DMS 1812197 and IIS 1816499.

References

John M Abowd. The US Census Bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2867–2867, 2018.

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

Hilal Asi and John C Duchi. Instance-optimality in differential privacy via approximate inverse sensitivity mechanisms. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14106–14117. Curran Associates, Inc., 2020a.

Hilal Asi and John C Duchi. Near instance-optimality in differential privacy. *arXiv preprint arXiv:2005.10630*, 2020b.

Jordan Awan and Aleksandra Slavković. Structure and sensitivity in differential privacy: Comparing k-norm mechanisms. *Journal of the American Statistical Association*, pages 1–20, 2020.

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

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

Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization, revisited. *rem*, 3:19, 2014b.

Stephen P Brooks, Yanan Fan, and Jeffrey S Rosenthal. Perfect forward simulation via simulated tempering. *Communications in Statistics-Simulation and Computation*, 35(3): 683–713, 2006.

Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

Mark Bun, Cynthia Dwork, Guy N Rothblum, and Thomas Steinke. Composable and versatile privacy via truncated CDP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 74–86, 2018.

Kamalika Chaudhuri, Anand Sarwate, and Kaushik Sinha. Near-optimal differentially private principal components. *Advances in Neural Information Processing Systems*, 25: 989–997, 2012.

Kamalika Chaudhuri, Anand D Sarwate, and Kaushik Sinha. A near-optimal algorithm for differentially-private principal components. *Journal of Machine Learning Research*, 14, 2013.

Karel Crombecq and Tom Dhaene. Generating sequential space-filling designs using genetic algorithms and Monte Carlo methods. In *Asia-Pacific Conference on Simulated Evolution and Learning*, pages 80–84. Springer, 2010.

Maria Daghofer and Wolfgang von der Linden. Perfect tempering. In *AIP Conference Proceedings*, volume 735, pages 355–362. American Institute of Physics, 2004.

Christos Dimitrakakis, Blaine Nelson, Zuhe Zhang, Aikateirni Mitrokotsa, and Benjamin Rubinstein. Differential privacy for Bayesian inference through posterior sampling. *Journal of Machine Learning Research*, 18(11):1–39, 2017.

Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *Advances in Neural Information Processing Systems 30*, December 2017.

Yevgeniy Dodis, Adriana López-Alt, Ilya Mironov, and Salil Vadhan. Differential privacy with imperfect randomness. In *Annual Cryptology Conference*, pages 497–516. Springer, 2012.

Jinshuo Dong, Aaron Roth, and Weijie J Su. Gaussian differential privacy. *Journal of the Royal Statistical Society Series B*, 84(1):3–37, 2022.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, pages 265–284. Springer, 2006.

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

Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 1054–1067, 2014.

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

Arun Ganesh and Kunal Talwar. Faster differentially private samplers via Rényi divergence analysis of discretized Langevin MCMC. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7222–7233. Curran Associates, Inc., 2020.

Simson L Garfinkel and Philip Leclerc. Randomness concerns when deploying differential privacy. In *Proceedings of the 19th Workshop on Privacy in the Electronic Society*, pages 73–86, 2020.

Andreas Haeberlen, Benjamin C Pierce, and Arjun Narayan. Differential privacy under fire. In *USENIX Security Symposium*, volume 33, 2011.

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

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

Christina Ilvento. Implementing the exponential mechanism with base-2 differential privacy. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 717–742, 2020.

G Joy Persial, M Prabhu, and R Shanmugalakshmi. Side channel attack-survey. *International Journal of Advanced Scientific Research and Review*, 1(4):54–57, 2011.

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

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

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

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

Luca Martino. *Independent random sampling methods*. Springer, 2018.

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

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

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

Ilya Mironov. On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 650–661, 2012.

Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

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

Jesper Møller and Geoff K Nicholls. *Perfect simulation for sample-based inference*. University of Aarhus. Centre for Mathematical Physics and Stochastics, 1999.

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

Luc Pronzato and Werner G Müller. Design of computer experiments: space filling and beyond. *Statistics and Computing*, 22(3):681–701, 2012.

James Propp and David Wilson. Coupling from the past: a user's guide. *Microsurveys in discrete probability*, 41:181–192, 1998.

Matthew Reimherr and Jordan Awan. KNG: The k-norm gradient mechanism. In *Advances in Neural Information Processing Systems*, pages 10208–10219, 2019.

Indrajit Roy, Srinath TV Setty, Ann Kilzer, Vitaly Shmatikov, and Emmett Witchel. Airavat: Security and privacy for mapreduce. In *NSDI*, volume 10, pages 297–312, 2010.

Jeremy Seeman, Matthew Reimherr, and Aleksandra Slavković. Exact privacy guarantees for markov chain implementations of the exponential mechanism with artificial atoms. *Advances in Neural Information Processing Systems*, 34:13125–13136, 2021.

Geoffrey Smith. On the foundations of quantitative information flow. In *International Conference on Foundations of Software Science and Computational Structures*, pages 288–302. Springer, 2009.

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

Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. Privacy loss in Apple's implementation of differential privacy on macOS 10.12. *arXiv preprint arXiv:1709.02753*, 2017.

Santosh Vempala and Andre Wibisono. Rapid convergence of the unadjusted Langevin algorithm: Isoperimetry suffices. In *Advances in Neural Information Processing Systems*, pages 8094–8106, 2019.

Sameer Wagh, Paul Cuff, and Prateek Mittal. Differentially private oblivious RAM. *Proceedings on Privacy Enhancing Technologies*, 4:64–84, 2018.

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

Yu-Xiang Wang, Stephen Fienberg, and Alex Smola. Privacy for free: Posterior sampling and stochastic gradient Monte Carlo. In *International Conference on Machine Learning*, pages 2493–2502, 2015.

Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

Larry Wasserman and Shuheng Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489):375–389, 2010.

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