New Oracle-Efficient Algorithms for Private Synthetic Data Release

Giuseppe Vietri [1] **Grace Tian** [2] **Mark Bun** [3] **Thomas Steinke** [4] **Steven Wu** [1]

Abstract

We present three new algorithms for constructing differentially private synthetic data—a sanitized version of a sensitive dataset that approximately preserves the answers to a large collection of statistical queries. All three algorithms are *oracle-efficient* in the sense that they are computationally efficient when given access to an optimization oracle. Such an oracle can be implemented using many existing (non-private) optimization tools such as sophisticated integer program solvers. While the accuracy of the synthetic data is contingent on the oracle's optimization performance, the algorithms satisfy differential privacy even in the worst case. For all three algorithms, we provide theoretical guarantees for both accuracy and privacy. Through empirical evaluation, we demonstrate that our methods scale well with both the dimensionality of the data and the number of queries. Compared to the state-of-the-art method High-Dimensional Matrix Mechanism (McKenna et al. VLDB 2018), our algorithms provide better accuracy in the large workload and high privacy regime (corresponding to low privacy loss ε).

1. Introduction

The wide range of personal data collected from individuals has facilitated many studies and data analyses that inform decisions related to science, commerce, and government policy. Since many of these rich datasets contain highly sensitive personal information, there is a tension between releasing useful information about the population and compromising individuals' privacy. In this work, we consider the problem of answering a large collection of statistical (or linear) queries subject to differential privacy constraints. Formally, we consider a data domain $\mathcal{X} = \{0,1\}^d$ of dimension d and a dataset $D \in \mathcal{X}^n$ consisting of the data of n individuals. Our goal is to approximately answer a large class of statistical queries $\mathcal{Q}$ about D. A predicate $\phi\colon \mathcal{X} \to [0,1]$ defines a statistical query, and the query $q_\phi : \mathcal{X}^n \to [0,1]$ is given by $q_\phi(D) = \frac{1}{n}\sum_{i=1}^n \phi(D_i)$. An approximate answer $a \in [0,1]$ must satisfy $|a - q_\phi(D)| \le \alpha$ for some accuracy parameter $\alpha > 0$. To preserve privacy, we work under the constraint of differential privacy (Dwork et al., 2006). Privately answering statistical queries is at the heart of the 2020 US Census release (Abowd, 2018) and provides the basis for a wide range of private data analysis tasks. For example, many machine learning algorithms can be simulated using statistical queries (Kearns, 1998).

An especially compelling way to perform private query release is to release *private synthetic data* – a sanitized version of the dataset that approximates all of the queries in the class $\mathcal{Q}$. Notable examples of private synthetic data algorithms are the SmallDB algorithm (Blum et al., 2008) and the private multiplicative weights (PMW) mechanism (Hardt & Rothblum, 2010) (and its more practical variant the multiplicative weights exponential mechanism MWEM (Hardt et al., 2012)), which can answer exponentially many queries and achieves nearly optimal sample complexity (Bun et al., 2018). Unfortunately, both algorithms involve maintaining a probability distribution over the data domain $\mathcal{X} = \{0,1\}^d$, and hence suffer exponential (in d) running time. Moreover, under standard cryptographic assumptions, this running time is necessary in the worst case (Ullman, 2016; Ullman & Vadhan, 2011). However, there is hope that these worst-case intractability results do not apply to real-world datasets.

To build more efficient solutions for constructing private synthetic data, we consider *oracle efficient* algorithms that rely on a black-box optimization subroutine. The optimization problem is NP-hard in the worst case. However, we invoke practical optimization heuristics for this subroutine (namely integer program solvers such as CPLEX and Gurobi). These heuristics work well on many real-world instances. Thus the algorithms we present are more practical than the worst-case hardness would suggest is possible. While our algorithms' efficiency and accuracy are contingent on the solver's performance, differential privacy is guaranteed even if the solver runs forever or fails to optimize correctly.

*Equal contribution [1]Department of Computer Science and Engineering, University of Minnesota [2]Harvard University [3]Boston University [4]IBM Research – Almaden. Correspondence to: Giuseppe Vietri <vietr002@umn.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

Overview of our results. To describe our algorithms, we will first revisit a formulation of the query release problem as a zero-sum game between a data player who maintains a distribution $\hat{D}$ over $\mathcal{X}$ and a query player who selects queries from $\mathcal{Q}$ (Hsu et al., 2013; Gaboardi et al., 2014). Intuitively, the data player aims to approximate the private dataset D with $\hat{D}$, while the query player tries to identify a query which distinguishes between D and $\hat{D}$. The prior work (Hsu et al., 2013; Gaboardi et al., 2014) showed that any (approximate) equilibrium for this game gives rise to an accurate synthetic dataset. To study the private equilibrium computation within this game, we consider a *primal* framework and a *dual* framework that enables us to unify and improve on existing algorithms.

In the primal framework, we perform the equilibrium computation via the following *no-regret dynamics*: over rounds, the data player updates its distribution $\hat{D}$ using a no-regret online learning algorithm, while the query player plays an approximate best response. The algorithm MWEM in prior work falls under the primal framework with the data player running the multiplicative weights (MW) method as the no-regret algorithm and the query player privately responding using the exponential mechanism (McSherry & Talwar, 2007). However, since the MW method maintains an entire distribution over the domain $\mathcal{X}$, MWEM runs in exponential time, even in the best case. To overcome this intractability, we propose two new algorithms FEM and sepFEM that follow the same no-regret dynamics but importantly replace the MW method with two variants of the follow-the-perturbed-leader (FTPL) algorithm (Kalai & Vempala, 2005)—Non-Convex-FTPL (Suggala & Netrapalli, 2019) and Separator-FTPL (Syrgkanis et al., 2016)—both of which solve a perturbed optimization problem instead of maintaining an exponential-sized distribution. FEM achieves an error rate of $\alpha = \tilde{O}(d^{3/4} \log^{1/2} |\mathcal{Q}|/n^{1/2})$, and sepFEM achieves a slightly better rate of $\alpha = \tilde{O}\left(d^{5/8} \log^{1/2} |\mathcal{Q}|/n^{1/2}\right)$, although the latter requires the query class $\mathcal{Q}$ to have a structure called a small separator set. In contrast, MWEM attains the error rate $\alpha = \tilde{O}\left(d^{1/4} \log^{1/2} |\mathcal{Q}|/n^{1/2}\right)$. Although the accuracy analysis requires repeated sampling from the FTPL distribution (and thus repeatedly solving perturbed integer programs), our experiments show that the algorithms remain accurate even with a much lower number of samples, which allows for a much more reasonable running time.

We then consider the *dual* formulation and improve upon the existing algorithm DualQuery (Gaboardi et al., 2014). Unlike MWEM, DualQuery has the query player running MW over the query class $\mathcal{Q}$, which is often significantly smaller than the data domain $\mathcal{X}$, and has the data player playing best response, which can be computed non-privately by solving an integer program. Since the query player's MW distribution is a function of the private data, DualQuery privately approximates this distribution with a collection of samples drawn from it. Each draw from the MW distribution can be viewed as a single instantiation of the exponential mechanism, which provides a bound on the privacy loss. We improve DualQuery by leveraging the observation that the MW distribution changes slowly between rounds in the no-regret dynamics. Thus can reuse previously drawn queries to approximate the current MW distribution via rejection sampling. By using this technique, our algorithm DQRS (DualQuery with rejection sampling) reduces the number of times we draw new samples from the MW distribution and also the privacy loss, and hence improves the privacy-utility trade-off. We theoretically demonstrate that DQRS improves the accuracy guarantee of DualQuery. Specifically DQRS attains accuracy $\alpha = \widetilde{O}\left(\frac{\log(|\mathcal{X}|/\beta)\cdot\log^3(|\mathcal{Q}|)}{n^2}\right)^{1/5}$ whereas DualQuery attains accuracy $\alpha = \widetilde{O}\left(\frac{\log(|\mathcal{X}|/\beta)\cdot\log^3(|\mathcal{Q}|)}{n^2}\right)^{1/6}$. Even though the dual algorithms DualQuery and DQRS have worse accuracy performance than the primal algorithms FEM and sepFEM, the dual algorithms run substantially faster since they make many fewer oracle calls. Thus we observe a trade-off not only between privacy and utility but also with computational resources.

In addition to our theoretical guarantees, we perform a basic experimental evaluation of our algorithms. As a benchmark, we use DualQuery as well as the state-of-the-art High-Dimensional Matrix Mechanism (HDMM) (McKenna et al., 2018); HDMM is being deployed in practice by the US Census Bureau (Kifer, 2019). We perform our experiments with the standard Adult and Loans datasets and use k-way conjunctions as a query workload. Our algorithms are comparable to the benchmarks on a small workload, and we see that FEM and HDMM performs best overall. We then compare HDMM and FEM on a large workload. Here we see that the accuracy of FEM is still similar to HDMM (and in some parameter regimes better). These results support our theoretical analysis.

In Section 6, we compare the performance of our algorithms against other practical algorithms for synthetic data generation. The benchmark we use is the High-Dimensional Matrix Mechanism (McKenna et al., 2018), which itself builds on the Matrix Mechanism (Li et al., 2015), but is more efficient and scalable. Given a workload of queries $\mathcal{Q}$, this algorithm uses optimization routines (in a significantly different way than ours) to select a different set of "strategy queries" which can be answered with Laplace noise. Answers to the original queries in $\mathcal{Q}$ can then be reconstructed by combining the noisy answers to these strategy queries.

2. Preliminaries

Definition 2.1 (Differential Privacy)**.** A randomized algorithm $\mathcal{M} : \mathcal{X}^* \to \mathcal{R}$ satisfies (ε, δ)-differential privacy if for all databases x, x' differing in at most one entry, and every measurable subset $S \subseteq \mathcal{R}$, we have

$$\Pr[\mathcal{M}(x) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(x') \in S] + \delta.$$

We will use the exponential mechanism as a key component in our design of private algorithms.

Definition 2.2 (Exponential Mechanism (McSherry & Talwar, 2007))**.** Given some database x, arbitrary range $\mathcal{R}$, and score function $S : \mathcal{X}^* \times \mathcal{R} \to \mathbb{R}$, the exponential mechanism $\mathcal{M}_E(x, S, \mathcal{R}, \varepsilon)$ selects and outputs an element $r \in \mathcal{R}$ with probability proportional to

$$\exp\left(\frac{\varepsilon S(x,r)}{2\Delta_S}\right),$$

where Δ_S is the sensitivity of S, defined as

$$\Delta_S = \max_{D,D':|D\triangle D'|=1, r\in R} |S(D,r) - S(D',r)|.$$

Lemma 1 ((McSherry & Talwar, 2007))**.** *The exponential mechanism $\mathcal{M}_E(x, S, \mathcal{R})$ is $(\varepsilon, 0)$-differentially private.*

Theorem 2 (Exponential Mechanism Utility (McSherry & Talwar, 2007))**.** *. Fixing a database x, let $OPT_S(x)$ denote the max score of function S. Then, with probability $1-\beta$ the error is bounded by:*

$$OPT_S(x) - S(x, \mathcal{M}_E(x, u, \mathcal{R}, \varepsilon)) \leq \frac{2\Delta_S}{\varepsilon}\left(\ln|\mathcal{R}|/\beta\right)$$

Theorem 3 (Advanced Composition (Dwork et al., 2010; Bun & Steinke, 2016))**.** *Let $\varepsilon, \delta, \delta' > 0$. The adaptive T-fold composition of T (ε, δ)-differentially private algorithms is $(\varepsilon', T\delta + \delta')$-differentially private for*

$$\varepsilon' = \frac{T\varepsilon^2}{2} + \varepsilon\sqrt{2T\log(1/\delta')}.$$

Finally, our algorithm will be using the following form of linear optimization oracle. In our experiments, we implement this oracle via an integer program solver.

Definition 2.3 (Linear Optimization Oracle)**.** Given as input a set of n statistical linear queries $\{q_i\}$ and a d-dimensional vector σ, a linear optimization oracle outputs

$$\hat{x} \in \arg\min_{x\in\{0,1\}^d} \left\{ \sum_{i=1}^{n} q_i(x) - \langle x, \sigma \rangle \right\}$$

3. Query Release Game

Given a class of queries $\mathcal{Q}$ over a database D, we want to output a differentially private synthetic dataset $\widehat{D}$ such that for any query $q \in \mathcal{Q}$ we have low error:

$$\text{error}(\widehat{D}) = |q(D) - q(\widehat{D})| \leq \alpha.$$

We revisit a zero-sum game formulation between a data-player and a query player for this problem (Hsu et al., 2013; Gaboardi et al., 2014). The data player has action set equal to the data universe $\mathcal{X}$ and the query player has action set equal to the query class $\mathcal{Q}$. We make the assumption that $\mathcal{Q}$ is closed under negation. That is, for every query $q \in \mathcal{Q}$ there is a *negated query* $\bar{q} \in \mathcal{Q}$ where $\bar{q}(D) = 1 - q(D)$. If $\mathcal{Q}$ is not closed under negation, we can simply add negated queries to $\mathcal{Q}$. Since $\mathcal{Q}$ is closed under negations, we can write the error as

$$|q(D) - q(\widehat{D})| = \max\{q(D) - q(\widehat{D}), \neg q(D) - \neg q(\widehat{D})\}$$

This allows us to define a payoff function that captures the error of $\widehat{D}$ without the absolute value. In particular, the payoff for actions $x \in \mathcal{X}$ and $q \in \mathcal{Q}$ is given by:

$$A(x,q) := q(D) - q(x) \tag{1}$$

The data player wants minimizes the payoff $A(x,q)$ while the query player maximizes it. Intuitively, the data player would like to find a distribution over $\mathcal{X}$ with low error, while the query player is trying to identify the query with the highest error. Each player chooses a mixed strategy, that is a distribution over their action set. Let $\Delta(\mathcal{X})$ and $\Delta(\mathcal{Q})$ denote the sets of distributions over $\mathcal{X}$ and $\mathcal{Q}$, respectively. For any $\widehat{D} \in \Delta(\mathcal{X})$ and $\widehat{Q} \in \Delta(\mathcal{Q})$, the payoff is defined as

$$A(\widehat{D}, \cdot) = \mathbb{E}_{x\sim\widehat{D}}\left[A(x,\cdot)\right], \quad A(\cdot, \widehat{Q}) = \mathbb{E}_{q\sim\widehat{Q}}\left[A(\cdot,q)\right].$$

A pair of mixed strategies $(\widehat{D}, \widehat{Q}) \in \Delta(\mathcal{X}) \times \Delta(\mathcal{Q})$ forms an α-approximate equilibrium of the game if

$$\max_{q\in\mathcal{Q}} A(\widehat{D}, q) - \alpha \leq A(\widehat{D}, \widehat{Q}) \leq \min_{x\in\mathcal{X}} A(x, \widehat{Q}) + \alpha, \tag{2}$$

The following result allows us to reduce the problem of query release to the problem of computing an equilibrium in the game.

Theorem 4 (Gaboardi et al. (2014))**.** *Let $(\widehat{D}, \widehat{Q})$ be any α-approximate equilibrium of the query release game, then the data player's strategy $\widehat{D}$ is 2α-accurate, that is for all $q \in \mathcal{Q}$, error$(\widehat{D}) = |q(D) - q(\widehat{D})| \leq 2\alpha$.*

3.1. No-Regret Dynamics

To compute such an equilibrium privately, we will simulate no-regret dynamics between the two players. Over rounds $t = 1, \ldots, T$, the two players will generate a sequence of plays $(D^1, Q^1), \ldots, (D^T, Q^T) \in \Delta(\mathcal{X}) \times \Delta(\mathcal{Q})$. The regrets of the two players are defined as

$$R_{\text{data}} = \sum_{t=1}^{T} A(D^t, Q^t) - \min_{x\in\mathcal{X}} \sum_{t=1}^{T} A(x, Q^t),$$

$$R_{\text{qry}} = \max_{q\in\mathcal{Q}} \sum_{t=1}^{T} A(D^t, q) - \sum_{t=1}^{T} A(D^t, Q^t)$$

Theorem 5 (Follows from (Freund & Schapire, 1997))**.** *The average play $(\overline{D}, \overline{Q})$ given by $\overline{D} = \frac{1}{T}\sum_{t=1}^{T} D^t$ and $\overline{Q} = \frac{1}{T}\sum_{t=1}^{T} Q^t$ from the no-regret dynamics above is an α-approximate equilibrium with*

$$\alpha = \frac{R_{data} + R_{qry}}{T}.$$

In the next section we will now provide a general framework to privately compute the approximate Nash equilibrium of the game.

4. Primal Oracle-Efficient Framework

In the primal framework, we will have the data player run a online learning algorithm to generate the distributions $D^1, \ldots, D^T$ over rounds and have the query player play an approximate best response Q^t against D^t in each round. The algorithm MWEM falls under this framework, but the no-regret algorithm (MW) runs in exponential time even in the best case since it maintains a distribution over the entire domain $\mathcal{X}$. We replace the MW method with two variants of the follow-the-perturbed-leader (FTPL) algorithm (Kalai & Vempala, 2005)—Non-Convex-FTPL (Suggala & Netrapalli, 2019) and Separator-FTPL (Syrgkanis et al., 2016). Both of these algorithms can generate a sample from their FTPL distributions by calling on an oracle to solve a perturbed optimization problem. (In our experiments, we instantiate this oracle with an integer program solver.) For both algorithms, the query player will select a query q_t (that is Q_t is point mass distribution on q_t) using the exponential mechanism. We present this primal framework in Algorithm 1.

Algorithm 1 Primal Framework of No-Regret Dynamics

Require: Target privacy ε, δ, target accuracy α, target failure probability β
input A dataset $D \in \mathcal{X}^n$, query class $\mathcal{Q}$, FTPL method $\mathcal{A}$, Number of rounds T
  Initialize ε_0 such that $\varepsilon = T\varepsilon_0^2/2 + \varepsilon_0\sqrt{2T\log(1/\delta)}$
  Let $\mathcal{M}_E$ be the exponential mechanism.
  for $t = 1$ **to** T **do**
    Data Player Generate D^t with online learner $\mathcal{A}$ with queries $q_1, \ldots, q_{t-1}$.
    Define quality function

$$S_t(D, q) = q(D) - q(\widehat{D^t})$$

    Query player Sample $q_t \sim \mathcal{M}_E(D, S_t, \mathcal{Q}, \varepsilon_0)$
  end for

Now we instantiate the primal framework above with two no-regret learners, which yield two algorithms FEM ((Non-Convex)-FTPL with Exponential Mechanism) and sepFEM (Separator-FTPL with Exponential Mechanism). First, the FEM algorithm at each round t computes a distribution D_t by solving a perturbed linear optimization problem polynomially many times. The optimization objective is given by the payoff against the previous queries and a linear perturbation

$$\arg\min_x \sum_{i=1}^{t-1} A(x, q_i) + \langle x, \sigma\rangle$$

where σ is a random vector drawn from the exponential distribution. Observe that the first term $q_i(D)$ in $A(x, q_i) = q_i(D) - q_i(x)$ does not depend on x. Thus, we can further simplify the objective as

$$\arg\max_x \left\{ \sum_{i=1}^{t-1} q_i(x) - \langle x, \sigma\rangle \right\}$$

To solve this problem above, we will use an linear optimization oracle (Definition 2.3), which we will implement using an integer program solver.

Algorithm 2 Data player update in FEM

Require: Exponential noise parameter η and number of samples s
  for $j \leftarrow 1$ **to** s **do**
    Let $\sigma_j \in \mathbb{R}^d$ be a random vector such that each coordinate of σ_j is drawn from the exponential distribution $\text{Exp}(\eta)$. Obtain a FTPL sample x_j^t by solving

$$\arg\max_x \left\{ \sum_{i=1}^{t-1} q_i(x) - \langle x, \sigma_j\rangle \right\}$$

  end for
output D_t as the uniform distribution over $\{x_1^t, \ldots, x_s^t\}$

The second algorithm is less general, but as we will show it achieves a better error rate for important classes of queries. Algorithm sepFEM relies on the assumption that the query class $\mathcal{Q}$ has a small separator set $\text{sep}(\mathcal{Q})$.

Definition 4.1 (Separator Set)**.** A set $\text{sep}(\mathcal{Q})$ is a separator set for queries $\mathcal{Q}$ if for any two distinct records $x, x' \in \mathcal{X}$, there exist $q \in \text{sep}(\mathcal{Q})$ such that $q(x) \neq q(x')$.

Many classes of statistical queries defined over the boolean hypercube have separator sets of size proportional to their VC-dimension or the dimension of the input data. For example, boolean conjunctions, disjunctions, halfspaces defined over the $\{0,1\}^d$, and parity functions all have separator sets of size d.

Algorithm sepFEM then perturbs the data player's optimization problem by inserting "fake" queries from the separator set:

$$\arg\max_x \left\{ \sum_{i=1}^{t-1} q_i(x) + \sum_{i=1}^{M} \sigma_{j,i} \tilde{q}_i(x) \right\},$$

where each $\sigma_{j,i}$ is a Laplace random weight. This problem can be viewed as a simple special case of the linear optimization problem in Definition 2.3 with no linear perturbation term.

Algorithm 3 Data player update in sepFEM

Require: Laplace noise parameters η and number of samples s, a separator set $\mathrm{sep}(\mathcal{Q}) = \{\tilde{q}_1, \ldots, \tilde{q}_M\}$ for $\mathcal{Q}$

for $i = 1$ **to** s **do**

Let $\sigma \in \mathbb{R}^M$ be a fresh random vector such that each coordinate of σ_j is drawn from the Laplace distribution $\mathrm{Lap}(\eta)$. Obtain a FTPL sample x_j^t by solving

$$\arg\max_x \left\{ \sum_{i=1}^{t-1} q_i(x) + \sum_{i=1}^{M} \sigma_{j,i} \tilde{q}_i(x) \right\}$$

end for

output D_t be a uniform distribution over $\{x_1^t, \ldots, x_s^t\}$

To derive the privacy guarantee of these two algorithms, we observe that the data player's update does not directly use the private dataset D. Thus, the privacy guarantee directly follows from the composition of T exponential mechanisms.

Theorem 6 (Privacy)**.** *Let $0 < \delta < 1$. For any no-regret algorithm $\mathcal{A}$, Algorithm 1 is (ε, δ)-differentially private.*

To derive the accuracy guarantee of the two algorithms, we first bound the regret of the two players. Note that the regret guarantee of the data player follow from the regret bounds on the two FTPL algorithms (Suggala & Netrapalli, 2019) and (Syrgkanis et al., 2016). The regret guarantee of the query player directly follows from the utility guarantee of the exponential mechanism (McSherry & Talwar, 2007). We defer the details to the appendix.

Corollary 6.1 (FEM Accuracy)**.** *With probability at least $1-\beta$, the algorithm FEM finds a synthetic database that answers all queries in $\mathcal{Q}$ with additive error*

$$\alpha = \widetilde{O}\left(\frac{d^{3/4} \log^{1/2} |\mathcal{Q}| \cdot \sqrt{\log(\frac{1}{\delta}) \log(\frac{1}{\beta})}}{n^{1/2} \varepsilon^{1/2}} \right)$$

Corollary 6.2 (sepFEM Accuracy)**.** *Let $\mathcal{Q}$ be a query class with a separator set $\mathrm{sep}(\mathcal{Q})$. With probability at least $1-\beta$, algorithm sepFEM finds a synthetic database that answers all queries in $\mathcal{Q}$ with additive error*

$$\alpha = O\left(\frac{|\mathrm{sep}(\mathcal{Q})|^{3/8} d^{1/4} \log^{1/2} |\mathcal{Q}| \cdot \sqrt{\log(\frac{1}{\delta}) \log(\frac{1}{\beta})}}{n^{1/2} \varepsilon^{1/2}} \right)$$

Note that if the query class $\mathcal{Q}$ has a separator set of size $O(d)$, which is the case for boolean conjunctions, disjunctions, halfspaces defined over the $\{0,1\}^d$, and parity functions, then the bound above becomes

$$\alpha = O\left(\frac{d^{5/8} \log^{1/2} |\mathcal{Q}| \cdot \sqrt{\log(\frac{1}{\delta}) \log(\frac{1}{\beta})}}{n^{1/2} \varepsilon^{1/2}} \right)$$

Remark. *Non-convex FEM and Separator FEM exhibit a better tradeoff between α and n than DualQuery, but a slightly worse dependence on d compared to DualQuery and MWEM.*

5. DQRS: DualQuery with Rejection Sampling

In this section, we present an algorithm DQRS that builds on the DualQuery algorithm (Gaboardi et al., 2014) and achieves better provable sample complexity. In DualQuery, we employ the dual framework of the query release game – the query player maintains a distribution over queries using the Multiplicative Weights (MW) no-regret learning algorithm and the data player best responds. However, the query player cannot directly use the distribution $\mathcal{Q}^t$ proposed by MW during round t because it depends on the private data. Instead, for each round t, it takes s samples from $\mathcal{Q}^t$ to form an estimate distribution $\widehat{\mathcal{Q}^t}$. The data player then best-responds against $\widehat{\mathcal{Q}^t}$. Sampling from the MW distribution $\mathcal{Q}^t$ can be interpreted as a sample from the exponential mechanism. The sampling step incurs a significant privacy cost.

Our algorithm DQRS improves the sampling step of DualQuery in order to reduce the privacy cost (and the runtime). The basic idea of our algorithm DQRS is to apply the rejection sampling technique to "recycle" samples from prior rounds. Namely, we generate some samples from $\mathcal{Q}^t$ using the samples obtained from the distribution in the previous round, i.e., $\mathcal{Q}^{t-1}$. This is possible because $\mathcal{Q}^t$ is close to $\mathcal{Q}^{t-1}$. We show that by taking fewer samples from $\mathcal{Q}^t$ for each round t, we consume less of the privacy budget. The result is that the algorithm operates for more iterations and obtains lower regret (i.e., better accuracy).

Theorem 7. DualQuery *with rejection sampling (Algorithm 4) takes in a private dataset $D \in \mathcal{X}^n$ and makes $T = O\left(\frac{\log|\mathcal{Q}|}{\alpha^2}\right)$ queries to an optimization oracle and outputs a dataset $\tilde{D} = (x^1, \cdots, x^T) \in \mathcal{X}^T$ such that, with probability at least $1-\beta$, for all $q \in \mathcal{Q}$ we have $|q(\tilde{D}) - q(D)| \le \alpha$. The algorithm is (ε, δ)-differentially private and attains accuracy*

$$\alpha = O\left(\frac{\log(|\mathcal{X}|T/\beta) \cdot \log^3(|\mathcal{Q}|) \cdot \log(1/\delta)}{n^2 \varepsilon^2} \right)^{1/5}.$$

In contrast, DualQuery (*without* rejection sampling) obtains the same result except with

$$\alpha = O\left(\frac{\log(|\mathcal{X}|T/\beta) \cdot \log^3(|\mathcal{Q}|) \cdot \log(1/\delta)}{n^2\varepsilon^2}\right)^{1/6}.$$

In other words, DQRS attains strictly better accuracy than DualQuery for the same setting of other parameters.

Algorithm 4 Rejection Sampling Dualquery

Require: Target accuracy $\alpha \in (0,1)$, target failure probability $\beta \in (0,1)$
input dataset D, and linear queries $q_1, \ldots, q_k \in \mathcal{Q}$
  Set $T = \frac{16\log|\mathcal{Q}|}{\alpha^2}$, $\eta = \frac{\alpha}{4}$
  $s = \frac{48\log(3|\mathcal{X}|T/\beta)}{\alpha^2}$
  Construct sample S_1 of s queries $\{q_i\}$ from $\mathcal{Q}$ according to $\mathcal{Q}^1 = \mathsf{Uniform}(\mathcal{Q})$
  for $t \leftarrow 1$ **to** T **do**
    Let $\tilde{q} = \frac{1}{s}\sum_{q \in S_t} q$
    Find x^t with $A_D(x^t, \tilde{q}) \geq \max_x A_D(x, \tilde{q}) - \alpha/4$
    Let $\gamma_t = \frac{1}{2t^{2/3}}$
    for all $q \in \mathcal{Q}$ **do**
      $\hat{\mathcal{Q}}_q^{t+1} := e^{-\eta-\gamma_t} \cdot \exp\left(-\eta A_D(x^t, q)\right) \mathcal{Q}_q^t$
    end for
    Normalize $\hat{\mathcal{Q}}^{t+1}$ to obtain $\mathcal{Q}^{t+1}$
    Construct S_{t+1} as follows
    Let $\tilde{s}_t = (2\gamma_t + 4\eta)s$ and add $\tilde{s}_t$ independent fresh samples from Q^{t+1} to S_{t+1}
    for all $q \in S_t$ **do**
      Add q to S_{t+1} with probability $\hat{\mathcal{Q}}_q^{\,t+1}/\mathcal{Q}_q^t$
      If $|S_{t+1}| > s$, discard elements at random so that $|S_{t+1}| = s$
    end for
  end for
output Sample $y_1, \ldots, y_s$

The analysis of DQRS largely follows that of DualQuery. The key difference is the analysis of the rejection sampling step, which is summarized by the following two lemmas. The first one shows that taking samples drawn from $Q = \mathcal{Q}^t$ and performing rejection sampling yields samples from $P = \mathcal{Q}^{t+1}$; thus S_{t+1} is distributed exactly as if it were drawn from $\mathcal{Q}^{t+1}$. The second lemma gives a bound on the privacy loss of the rejection sampling step.

Lemma 8 (Rejection Sampling Accuracy)**.** *Let P and Q be probability distributions over $\mathcal{Q}$, and let $M \geq \max_{q\in\mathcal{Q}} P_q/Q_q$. Sample an element of $\mathcal{Q}$ as follows. Sample q according to Q, and accept it with probability $P_q/(M \cdot Q_q)$. If q is not accepted, sample q according to P. Then the resulting element is distributed according to P.*

Lemma 9 (Rejection Sampling Privacy)**.** *The subroutine which accepts q with probability $\hat{Q}_q^{t+1}/Q_q^t = e^{-\eta-\gamma_t} \cdot \exp(-\eta A_D(x^t, q))$ is ε-differentially private for $\varepsilon = \max\{\eta/n, \eta/\gamma_t n\}$.*

6. Experiments on the Adult dataset

We evaluate the algorithms presented in this paper on two different datasets: the ADULT dataset from the UCI repository (Dua & Graff, 2017) and the LOANS dataset. We focus on answering 5-way marginals, with privacy budget ε taking value in $0.1, 0.15, 0.2, 0.25, 0.5$ and 1. The datasets in our experiments are summarized in table 1. To measure the accuracy of a synthetic dataset $\widehat{d}$ we used the max additive error over a set of queries $\{q_j\}$:

$$\mathrm{err}(\widehat{D}) = \max_j \left| q_j(D) - q_j(\widehat{D}) \right|$$

In our implementation, algorithm FEM has hyperparameters ε_0 and η controlling the accuracy and the run time of the algorithm. We ran FEM on 20 different hyperparameter combinations and report the one with the minimum error in figures 1, 2, 3. The the range of hyperparameter we used is summarized in tables 2 and 3.

Table 1. Datasets

DATA SET	RECORDS	ATTRIBUTES
ADULT	48842	15
LOANS	42535	48

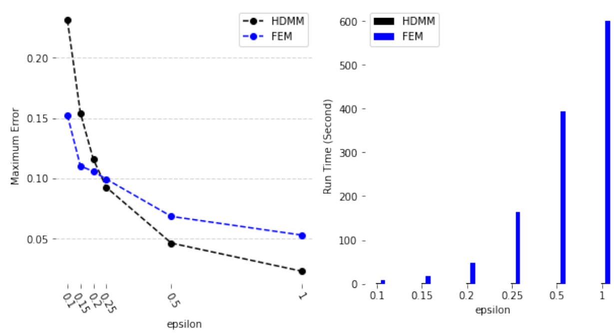

Figure 1. The ADULT dataset with 48,842 rows and 15 attributes. Running with The queries consist of $1,213,952$ 5-way marginals.

We observed that among our three algorithms, FEM consistently gets the best accuracy across different values of ε. Therefore we focus on comparing FEM against HDMM, which is a state-of-the-art method. As shown in figure 1 and 2, FEM scales better to the high privacy regime (low ε value). However, HDMM has better runtime. From figure 3, we also see that FEM scales much better when the workload size if very large.

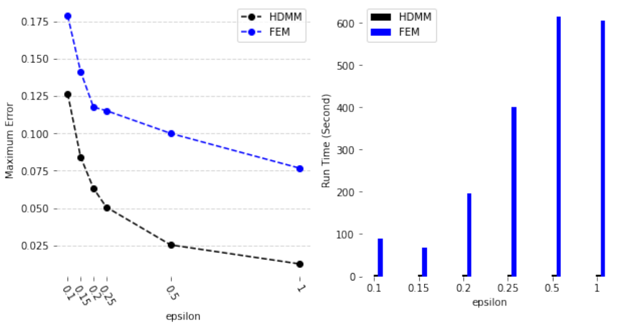

Figure 2. The LOANS dataset with $42,535$ rows and 48 attributes. The queries consist of $588,584$ 5-way marginals.

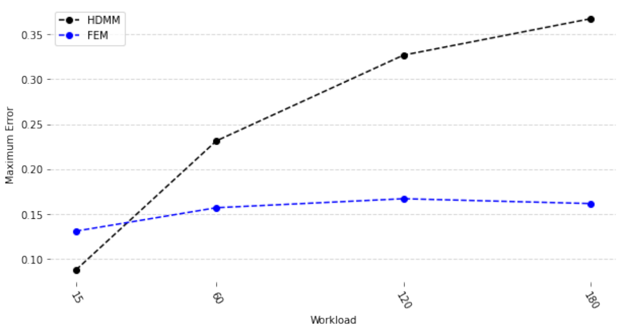

Figure 3. Maximum error on the ADULT data set for different workload sizes. The workload sizes are 15, 60, 120, 180. The queries are the 5-way marginals.

Optimazing over k-way Marginals In our experiments, we optimize over a set of k-way marginal queries. On each round t the algorithm FEM takes as input a sequence of t queries $(q_1, \dots, q_t)$ and a random perturbation term σ and solves the following optimization problem

$$\arg\max_x \left\{ \sum_{i=1}^{t-1} q_i(x) - \langle x, \sigma \rangle \right\}$$

We represent a data record by its one-hot binary encoding with dimension d, i.e. $x \in \{0,1\}^d$. The i^{th} k-way marginal query q_i is associated with a binary vector $w_i \in \{0,1\}^d$ with $\sum_{i=1}^d w_i = k$ which is an encoding of a conjunction over atributes in x, such that

$$q_i(x) = \begin{cases} 1 & \text{if } k = \sum_{j=1}^d x_j w_{i,j} \\ 0 & \text{otherwise} \end{cases}$$

If q_i is a negated query then

$$q_i(x) = \begin{cases} 0 & \text{if } k = \sum_{j=1}^d x_j w_{i,j} \\ 1 & \text{otherwise} \end{cases}$$

To implement the linear optimization oracle, we encode the optimization problem as the following integer program

$$\max \sum_{i<t} c_i - \sum_{j=1}^d x_j \sigma_j$$

$$\text{s.t.} \sum_{j=1}^d x_j w_{i,j} \geq k c_i \quad \text{if } q_i \text{ is not negated}$$

$$\sum_{j=1}^d (1 - x_j) w_{i,j} \geq c_i \quad \text{if } q_i \text{ is negated}$$

We used the Gurobi solver for mixed-integer-programming to solve the integer program.

Parameter Tunning In our implementation, algorithm FEM has hyperparameters ε_0 and η. Both the accuracy and the run time of the algorithm depend on how we choose these hyperparameters. Table 2 contains a summary of FEM's parameters and the range of values that we tested on. The η parameter is the scale of the random objective perturbation term. The data player samples a synthetic dataset $\widehat{D}$ from the Follow The Perturbed Leader distribution with parameter η as in algorithm 2. The perturbation scale η controls the rate of convergence of the algorithm. Setting this value too low can make the algorithm unstable and leads to bad performance. If set too high, the solver in FTPL focuses too much on optimizing over the noise term.

The parameter ε_0 corresponds to the privacy consumed on each round by the exponential mechanism with the parameter ε_0. The goal is to find a query that maximizes the error on $\widehat{D}$. Thus, the parameter ε_0 controls the number of iterations. Again we face a trade-off in choosing ε_0, since setting this value too high can lead to too few iterations giving the algorithm no chance to converge to a good solution. If ε_0 is too low, it can make the algorithm run too slow, and also it makes it hard for the query player's exponential mechanism to find queries with large errors.

Table 2. FEM hyperparameters.

PARAM	DESCRIPTION	RANGE
ε_0	PRIVACY BUDGET USED PER ROUND	0.001, 0.002, 0.003, 0.004, 0.1
η	SCALE OF NOISE FOR OBJECTIVE PERTURBATION	1, 2, 3, 4

For algorithm DualQuery and DQRS we set parameters η and s, where η is the update rate of MW and s is the number of samples from the distribution of MW. Table 3 summarizes the range of parameters that we tried.

Table 3. DualQuery and DQRS hyperparameters.

PARAM	DESCRIPTION	RANGE
η	MULTIPLICATIVE WEIGHT RATE	0.2, 0.3, 0.4, 0.5, 0.6, 0.7
s	NUMBER OF SAMPLES FROM MW DISTRIBUTION	10, 20, 30, 40, 50, 100

The implementation We ran the experiments on a machine with 4-core Opteron processor and 192 Gb of ram.

7. Conclusion and Future Work

In this paper, we have studied the pressing problem of efficiently generating private synthetic data. We have presented three new algorithms for this task that sidestep known worst-case hardness results by using heuristic solvers for NP-complete subroutines. All of our algorithms are equipped with formal privacy and utility guarantees and they are oracle-efficient – i.e., our algorithms are efficient as long as the heuristic solvers are efficient.

There is a very real need for practical private synthetic data generation tools and a dearth of solutions available; the scientific literature offers mostly exponential-time algorithms and negative intractability results. This work explores one avenue for solving this conundrum and we hope that there is further work both extending this line of work and exploring entirely new approaches. Our experimental evaluation demonstrates that our algorithms are promising and supports our theoretical results. However, our experiments are relatively rudimentary. In particular, we invested most time into optimizing the most promising algorithm FEM. An immediate question is whether further optimization of the other two algorithms could yield better results.

References

Abowd, J. M. The U.S. census bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pp. 2867, 2018. doi: 10.1145/3219819.3226070. URL https://doi.org/10.1145/3219819.3226070.

Blum, A., Ligett, K., and Roth, A. A learning theory approach to non-interactive database privacy. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pp. 609–618, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-047-0. doi: 10.1145/1374376.1374464. URL http://doi.acm.org/10.1145/1374376.1374464.

Bun, M. and Steinke, T. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Proceedings of the 14th Conference on Theory of Cryptography*, TCC '16-B, pp. 635–658, Berlin, Heidelberg, 2016. Springer.

Bun, M., Ullman, J., and Vadhan, S. P. Fingerprinting codes and the price of approximate differential privacy. *SIAM J. Comput.*, 47(5):1888–1938, 2018. doi: 10.1137/15M1033587. URL https://doi.org/10.1137/15M1033587.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Conference on Theory of Cryptography*, TCC '06, pp. 265–284, Berlin, Heidelberg, 2006. Springer.

Dwork, C., Rothblum, G. N., and Vadhan, S. P. Boosting and differential privacy. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pp. 51–60. IEEE Computer Society, 2010.

Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139, 1997. ISSN 0022-0000. doi: https://doi.org/10.1006/jcss.1997.1504. URL http://www.sciencedirect.com/science/article/pii/S002200009791504X.

Gaboardi, M., Arias, E. J. G., Hsu, J., Roth, A., and Wu, Z. S. Dual query: Practical private query release for high dimensional data. In *International Conference on Machine Learning*, pp. 1170–1178, 2014.

Hardt, M. and Rothblum, G. N. A multiplicative weights mechanism for privacy-preserving data analysis. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pp. 61–70. IEEE, 2010.

Hardt, M., Ligett, K., and McSherry, F. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems*, pp. 2339–2347, 2012.

Hsu, J., Roth, A., and Ullman, J. Differential privacy for the analyst via private equilibrium computation. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pp. 341–350. ACM, 2013.

Kalai, A. T. and Vempala, S. Efficient algorithms for online decision problems. *J. Comput. Syst. Sci.*, 71(3):291–307, 2005. doi: 10.1016/j.jcss.2004.10.016. URL https://doi.org/10.1016/j.jcss.2004.10.016.

Kearns, M. J. Efficient noise-tolerant learning from statistical queries. *J. ACM*, 45(6):983–1006, 1998. doi: 10.1145/293347.293351. URL https://doi.org/10.1145/293347.293351.

Kifer, D. Consistency with external knowledge: The topdown algorithm, 2019. http://www.cse.psu.edu/~duk17/papers/topdown.pdf.

Li, C., Miklau, G., Hay, M., McGregor, A., and Rastogi, V. The matrix mechanism: optimizing linear counting queries under differential privacy. *VLDB J.*, 24(6):757–781, 2015.

McKenna, R., Miklau, G., Hay, M., and Machanavajjhala, A. Optimizing error of high-dimensional statistical queries under differential privacy. *PVLDB*, 11(10):1206–1219, 2018.

McSherry, F. and Talwar, K. Mechanism design via differential privacy. In *FOCS*, volume 7, pp. 94–103, 2007.

Suggala, A. S. and Netrapalli, P. Online non-convex learning: Following the perturbed leader is optimal. *CoRR*, abs/1903.08110, 2019. URL http://arxiv.org/abs/1903.08110.

Syrgkanis, V., Krishnamurthy, A., and Schapire, R. E. Efficient algorithms for adversarial contextual learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 2159–2168. JMLR.org, 2016. URL http://dl.acm.org/citation.cfm?id=3045390.3045618.

Ullman, J. Answering $n^{2+o(1)}$ counting queries with differential privacy is hard. *SIAM J. Comput.*, 45(2):473–496, 2016. doi: 10.1137/130928121. URL https://doi.org/10.1137/130928121.

Ullman, J. and Vadhan, S. Pcps and the hardness of generating private synthetic data. In *Theory of Cryptography Conference*, pp. 400–416. Springer, 2011.