SPPM: Sparse Privacy Preserving Mappings

Salman Salamatian [*] **Nadia Fawaz** [†] **Branislav Kveton** [†] **Nina Taft** [†]
[*] École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland
[†] Technicolor, USA

Abstract

We study the problem of a user who has both public and private data, and wants to release the public data, e.g. to a recommendation service, yet simultaneously wants to protect his private data from being inferred via big data analytics. This problem has previously been formulated as a convex optimization problem with linear constraints where the objective is to minimize the mutual information between the private and released data. This attractive formulation faces a challenge in practice because when the underlying alphabet of the user profile is large, there are too many potential ways to distort the original profile. We address this fundamental scalability challenge. We propose to generate sparse privacy-preserving mappings by recasting the problem as a sequence of linear programs and solving each of these incrementally using an adaptation of Dantzig-Wolfe decomposition. We evaluate our approach on several datasets and demonstrate that nearly optimal privacy-preserving mappings can be learned quickly even at scale.

1 Introduction

Finding the right balance between privacy risks and big data rewards is one of the biggest challenges facing society today. Big data creates tremendous opportunity, especially for all of the services that offer personalized advice. Recommendation services are rampant today and offer advice on everything including movies, TV shows, restaurants, music, sleep, exercise, vacation, entertainment, shopping, and even friends. On one hand, people are willing to part with some of their personal data (e.g. movie watching history) for the sake of these services. On the other hand, many users have some data about themselves they would prefer to keep private (e.g. their political affiliation, salary, pregnancy status, religion). Most individuals have both public and private data and hence they need to maintain a boundary between these different elements of their personal information. This is an enormous challenge because inference analysis on publicly released data can often uncover private data [27, 6, 21].

A number of research efforts have explored the idea of distorting the released data [29, 26, 17, 7, 11, 3] to preserve user's privacy. In some of these prior efforts, distortion aims at creating some confusion around user data by making its value hard to distinguish from other possible values; in other efforts, distortion is designed to counter a particular inference threat (i.e. a specific classifier or analysis). Recently [7] proposed a new framework for data distortion, based on information theory, that captures privacy leakage in terms of mutual information. Minimizing the mutual information between a user's private data and released data is attractive because it reduces the correlation between the private data and the publicly released data, and thus any inference analysis that tries to learn the private data from the public data is rendered weak, if not useless. In other words, this approach is agnostic to the type of inference analysis used in a given threat.

This promising framework, while theoretically sound, faces some challenges in terms of bridging the gap between theory and practicality. Distorting data, in the context of recommendation systems, means altering a user's profile. The framework casts this privacy problem as a convex optimization problem with linear constraints, where the number of variables grows quadratically with the size of the underlying alphabet that describes user's profiles. In real world systems, the alphabet can be huge, thus the enormous number of options for distorting user profiles presents a scalability challenge, that we address in this paper.

We make two contributions to handle scalability. First, by studying small scale problems, both analytically and empirically, we identify that mappings to distort profiles are in fact naturally sparse. We leverage this observation to develop sparse privacy-preserving mappings (SPPM). Second, we propose an algorithm, called SPPM that handles scalability through two insights. Although the underlying optimization problem has linear constraints, its objective function is nonlinear. We use the Frank-Wolfe algorithm that approximates the objective via a sequence of linear approximations that can be solved quickly. To do this, we adapt the Dantzig-Wolfe decomposition to the structure of our problem. Overall we reduce the number of variables from quadratic to linear in the number of user profiles. To the best of our knowledge, this work is the first to apply large scale linear programming optimization techniques to privacy problems.

A salient feature of our novel approach is that the distortion applied to each user is personalized, meaning that it is tailored for that user. Our solution does not require applying distortion to profiles in a uniform way, nor requires any monotonistic behavior such as mapping a profile to a far neighbor with decreasing likelihood (such as in [19]. We will see that the flexibility our system allows for a better use of the distortion budget where it is needed, and avoids wasting this budget on unnecessary distortions.

Our third contribution is a detailed evaluation on three datasets, in which we compare our solution to an optimal one (when feasible) and to a state-of-the-art solution based on differential privacy (called the Exponential Mechanism [19]). We find that our solutions are close to optimal, and consistently outperform the exponential mechanism (ExpMec) approach in that we achieve more privacy with less distortion. We show that our methods scale well with respect to the number of user profiles and their underlying alphabet.

Related Work. We consider the framework for privacy against statistical inference in [7, 18, 24]. In [24], a method based on quantization was proposed to reduce the number of optimization variables. It was shown that the reduction in complexity does not affect the privacy levels that can be achieved, but comes at the expense of additional distortion. In [18], privacy mappings in the class of parametric additive noise were considered, which allow the number of optimization variables to be reduced to the number of noise parameters. However, this suboptimal solution is not suitable for perfect privacy, as it requires a high distortion. In this paper, we propose to exploit the structure of the optimization to achieve computational speed-ups that will allow scalability. To the best of our knowledge, this is the first paper that evaluates the privacy-utility framework in [7] on such a large scale.

Our use of the information theoretic framework [7] relies on a *local privacy* setting, where users do not trust the entity collecting data, thus each user holds his data locally, and passes it through a privacy-preserving mechanism before releasing it to the untrusted entity. Local privacy dates back to randomized response in surveys [28], and has been considered in privacy for data mining and statistics [2, 12, 20, 16, 4, 7, 18, 9]. Information theoretic privacy metrics have also been considered in [23, 29, 12, 22, 25]. Finally, differential privacy [10] is currently the prevalent notion of privacy in privacy research. In particular, the exponential mechanism, to which we compare our privacy mapping in Section 4.3, was introduced in [19].

The general problem of minimizing a convex function under convex constraints has been studied extensively, and is of crucial importance in many machine learning tasks. The idea of a sparse approximate solutions to those problems has also been studied in the literature and is often called *Sparse Greedy Approximation* [8, 15, 14, 13, 30]. This type of algorithm has been used with success in many applications such as Neural Network [15], Matrix Factorization [14], SVM [13], Boosting [30], etc. We apply this approach to a new problem and adapt it to efficiently handle our scalability challenges. Another common approach to minimizing a convex function is stochastic gradient descent [31]. This approach is preferable when the feasible set has a simple form and is easy to project to. In our case, the feasible set is defined by many constraints, thus we opted for the Frank-Wolfe algorithm.

2 Problem Statement and Challenges

Privacy-Utility Framework: We consider the setting described in [7], where a user has two kinds of data: a vector of personal data $A \in \mathcal{A}$ that he would like to remain private, e.g. his income level, his political views, and a vector of data $B \in \mathcal{B}$ that he is willing to release publicly in order to receive a useful service (such as releasing his media preferences to a recommender service to receive content recommendations). $\mathcal{A}$ and $\mathcal{B}$ are the sets from which A and B can assume values. We assume that the user private attributes A are linked to his data B by the joint probability distribution $p_{A,B}$. Thus, an adversary who would observe B could infer some information about A. To reduce this inference threat, instead of releasing B, the user releases a *distorted version* of B, denoted $\hat{B} \in \hat{\mathcal{B}}$, generated according to a conditional probabilistic mapping $p_{\hat{B}|B}$, called the *privacy-preserving mapping*. Note that the set $\hat{\mathcal{B}}$ may differ from $\mathcal{B}$. This setting is reminiscent of the *local privacy* setting (e.g. randomized response, input perturbation) [28], where users do not trust the entity collecting data, thus each user holds his data locally, and passes it through a privacy-preserving mechanism before releasing it. The privacy mapping $p_{\hat{B}|B}$ is designed to render any statistical inference of A based on the observation of $\hat{B}$ harder, while preserving some utility to the released data $\hat{B}$, by limiting the distortion caused by the mapping. Following the framework for privacy-utility against statistical inference in [7], the inference threat is modeled by the mutual information $I(A; \hat{B})$ between the private attributes A and the publicly released data $\hat{B}$, while the utility requirement is modeled by a constraint on the average distortion $E_{B,\hat{B}}[d(B, \hat{B})] \leq \Delta$, for some distortion metric $d : \mathcal{B} \times \hat{\mathcal{B}} \to \mathbb{R}^+$, and $\Delta \geq 0$.

In the case of perfect privacy ($I(A;\hat{B}) = 0$), the privacy mapping $p_{\hat{B}|B}$ renders the released data $\hat{B}$ statistically independent from the private data A.

In general, a system that provides privacy protection does not know in advance the inference algorithm that the adversary will run on released data. This framework is appealing because it works regardless of the inference algorithm used by an adversary. In addition, this framework allows for the use of any distortion metric, either generic metrics such as Hamming, l_p, or weighted norms, or specific utility metrics tailored to a given learning algorithm that will run on the released user data. The use of a generic distance in the utility constraint is relevant in several cases. First, the learning algorithm that will run on the released data may not be known in advance to the system providing privacy protection, as it may be proprietary information that belongs to the service provider. Second, the released user data may be used by numerous analyses tasks based on multiple different ML algorithms, in which case the utility constraint should not be limited to one specific distortion metric.

Both the mutual information $I(A;\hat{B})$ and the average distortion $E_{B,\hat{B}}[d(B,\hat{B})]$ depend on both the prior distribution $p_{A,B}$ and the privacy mapping $p_{\hat{B}|B}$, since $A \to B \to \hat{B}$ form a Markov chain. To stress these dependencies, we will denote $I(A;\hat{B}) = J(p_{A,B}, p_{\hat{B}|B})$. Consequently, given a prior $p_{A,B}$ linking the private attributes A and the data B, the privacy mapping $p_{\hat{B}|B}$ minimizing the inference threat subject to a distortion constraint is obtained as the solution to the following convex optimization problem [7] [1]

$$\begin{aligned} &\underset{p_{\hat{B}|B} \in \text{Simplex}}{\text{minimize}} \quad J(p_{A,B}, p_{\hat{B}|B}) \\ &\text{s.t. } \mathbb{E}_{p_{B,\hat{B}}}\left[d(B,\hat{B})\right] \leq \Delta, \end{aligned} \tag{1}$$

where Simplex denotes the probability simplex ($\sum_x p(x) = 1,\ \ p(x) \geq 0\ \forall x$).

Sparsity of the privacy mapping: When applying the aforementioned privacy-accuracy framework to large data, we encounter a challenge of scalability. Designing the privacy mapping requires characterizing the value of $p_{\hat{B}|B}(\hat{b}|b)$ for all possible pairs $(b,\hat{b}) \in \mathcal{B} \times \hat{\mathcal{B}}$, i.e. solving the convex optimization problem over $|\mathcal{B}||\hat{\mathcal{B}}|$ variables. When $\hat{\mathcal{B}} = \mathcal{B}$, and the size of the alphabet $|\mathcal{B}|$ is large, solving the optimization over $|\mathcal{B}|^2$ variables may become intractable.

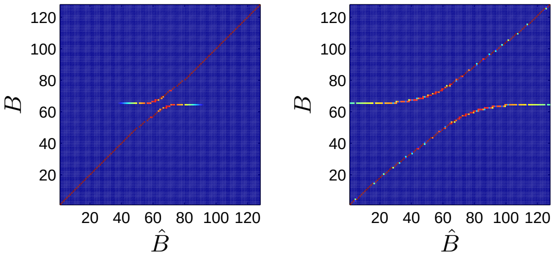

Figure 1: Optimal mappings for Toy example

A natural question is whether $\hat{\mathcal{B}} = \mathcal{B}$ is a necessary assumption to achieve the optimal privacy-utility tradeoff. In other words, does Optimization (1) need to be solved over $|\mathcal{B}|^2$ variables to achieve optimality? The following theoretical toy example motivates a sparser approach to the design of the privacy mapping.

Example: Let $A \in \{0,1\}$, and $B \in \{1,2,\dots,2^m\}$, and define the joint distribution $p_{A,B}$ such that $p(A = 0) = p(A = 1) = \frac{1}{2}$, and for $i \in \{1,2,\dots,2^m\}$, let $p(B = i|A = 0) = \frac{1}{2^{m-1}}$ if $i \leq 2^{m-1}$, 0 otherwise, and let $p(B = i|A = 1) = \frac{1}{2^{m-1}}$ if $i > 2^{m-1}$, 0 otherwise. For this example, the privacy threat is the worse it could be, as observing B determines deterministically the value of the private random variable A (equivalently, $I(A;B) = H(A) = 1$). In Fig. 2, we consider an l_2 distortion measure $d(B;\hat{B}) = (B-\hat{B})^2$ and illustrate the optimal mappings solving Problem ((1)) for different distortion values. For small distortions, the red diagonal in Figure 2 shows that most points $B = b$ are only mapped to themselves $\hat{B} = b$. As we increase the distortion level, each point $B = b$ gets mapped to a larger number of points $\hat{B} = \hat{b}$.

This theoretical example, as well as experiments on real world datasets such as the census data have shown that the optimal privacy preserving mapping may turn out to be sparse, in the sense that the support of $p_{\hat{B}|B}(\hat{b}|b)$ may be of much smaller size than $\mathcal{B}$, and may differ for different values of $B = b$. We propose to exploit sparsity properties of the privacy mapping to speed up the computation, by picking the support of $p_{\hat{B}|B}(\hat{b}|b)$, i.e. the set of points $\hat{B} = \hat{b}$ to which $B = b$ can be mapped with a non-zero probability, in an iterative greedy way. Although we a priori motivate the sparse approach using one theoretical example, and limited empirical evidence from some datasets, our experimental results demonstrate, a posteriori, that the sparse approach performs close to optimally on other datasets, and thus justifies empirically its relevance.

3 Sparse and Greedy Algorithm

Before we describe our algorithm, we rewrite Optimization (1) compactly. Let $\mathbf{X}$ be a $n \times n$ matrix of optimized variables, whose entries are defined as

[1] Solving Optimization (1) for different values of Δ allows to generate the whole privacy-utility curve, as in Fig. 2. Any point on or above this curve is achievable. Using this curve, one can efficiently solve the related problem of maximizing utility given a constraint on privacy: for a given privacy requirement, the curve gives the operating point corresponding to the smallest loss in utility.

$x_{i,j} = p_{\hat{B}|B}(\hat{b}_i \mid b_j)$, and let $\mathbf{x}_j$ be the j-th column of $\mathbf{X}$. To highlight the optimization aspect of our problem, we write the objective function $J(p_{A,B}, p_{\hat{B}|B})$ as a function $f(\mathbf{X})$, with the understanding that f depends on $p_{A,B}$, which is not optimized, and on $\mathbf{X}$, which is optimized. Similarly the distortion constraint can be written as $\sum_{j=1}^{n} \mathbf{d}_j^\intercal \mathbf{x}_j \leq \Delta$, where each $\mathbf{d}_j = p_B(b_j)(d(\hat{b}_1, b_j),\ d(\hat{b}_2, b_j),\ \ldots,\ d(\hat{b}_n, b_j))^\intercal$ is a vector of length n that represents the distortion metric scaled by the probability of the corresponding symbol b_j. The marginal of B is computed as $p_B(b_j) = \sum_a p_{A,B}(a, b_j)$. Finally, the simplex constrain can be written as $\mathbf{1}_n \mathbf{x}_j = 1$ for all j, where $\mathbf{1}_n$ is an all-ones vector of length n. Given the new notation, our original problem (1) can be written compactly as:

$$\begin{aligned} \underset{\mathbf{X}}{\text{minimize}} \quad & f(\mathbf{X}) \\ \text{subject to} \quad & \sum_{j=1}^{n} \mathbf{d}_j^\intercal \mathbf{x}_j \leq \Delta \\ & \mathbf{1}_n^\intercal \mathbf{x}_j = 1 \quad \forall j = 1, \ldots, n \\ & \mathbf{X} \geq 0 \end{aligned} \tag{2}$$

where $\mathbf{X} \geq 0$ is an entry-wise inequality.

3.1 Franke-Wolfe Linearization

The optimization problem (1) has linear constraints but its objective function is non-linear. In this paper, we solve the problem as a sequence of linear programs, also known as the *Frank-Wolfe method.* Each iteration ℓ of the method consists of three major steps. First, we compute the gradient $\nabla_{\mathbf{X}} f(\mathbf{X}_{\ell-1})$ at the solution from the previous step $\mathbf{X}_{\ell-1}$. The gradient is a $n \times n$ matrix $\mathbf{C}$, where $c_{i,j} = \frac{\partial}{\partial x_{i,j}} f(\mathbf{X}_{\ell-1})$ is a partial derivative of the objective function with respect to the variable $x_{i,j}$. Second, we find a feasible solution $\mathbf{X}'$ in the direction of the gradient. This problem is solved as a linear program with the same constraint as the original problem:

$$\begin{aligned} \underset{\mathbf{X}}{\text{minimize}} \quad & \sum_{j=1}^{n} \mathbf{c}_j^\intercal \mathbf{x}_j \\ \text{subject to} \quad & \sum_{j=1}^{n} \mathbf{d}_j^\intercal \mathbf{x}_j \leq \Delta \\ & \mathbf{1}_n^\intercal \mathbf{x}_j = 1 \quad \forall j = 1, \ldots, n \\ & \mathbf{X} \geq 0 \end{aligned} \tag{3}$$

where $\mathbf{c}_j$ is the j-th column of $\mathbf{C}$. Finally, we find the minimum of f between $\mathbf{X}_{\ell-1}$ and $\mathbf{X}'$, $\mathbf{X}_\ell$, and make it the current solution. Since f is convex, this minimum can be found efficiently by ternary search. The minimum is also feasible because the feasible region is convex, and both $\mathbf{X}'$ and $\mathbf{X}_{\ell-1}$ are feasible.

Algorithm 1 SPPM: Sparse privacy preserving maps

```
Input: Starting point $\mathbf{X}_0$, number of steps $L$

for all $\ell = 1, 2, \ldots, L$ do
    $\mathbf{C} \leftarrow \nabla_{\mathbf{X}} f(\mathbf{X}_{\ell-1})$
    $\mathcal{V} \leftarrow$ DWD
    Find a feasible solution $\mathbf{X}'$ in the direction of the gradient $\mathbf{C}$:

        minimize_X   $\sum_{j=1}^{n} \mathbf{c}_j^\intercal \mathbf{x}_j$                     (4)
        subject to   $\sum_{j=1}^{n} \mathbf{d}_j^\intercal \mathbf{x}_j \leq \Delta$
                     $\mathbf{1}_n^\intercal \mathbf{x}_j = 1 \quad \forall j = 1, \ldots, n$
                     $\mathbf{X} \geq 0$
                     $x_{i,j} = 0 \quad \forall (i,j) \notin \mathcal{V}$

    Find the minimum of $f$ between $\mathbf{X}_{\ell-1}$ and $\mathbf{X}'$:

        $\gamma^* \leftarrow \arg\min_{\gamma \in [0,1]} f((1-\gamma)\mathbf{X}_{\ell-1} + \gamma \mathbf{X}')$      (5)

    $\mathbf{X}_\ell \leftarrow (1-\gamma^*)\mathbf{X}_{\ell-1} + \gamma^* \mathbf{X}'$

Output: Suboptimal feasible solution $\mathbf{X}_L$
```

3.2 Sparse Approximation

The linear program (3) has n^2 variables and therefore is hard to solve when n is large. In this section, we propose an incremental solution to this problem, which is defined only on a subset of *active variables* $\mathcal{V} \subseteq \{1, 2, \ldots, n\} \times \{1, 2, \ldots, n\}$. The active variables are the non-zero variables in the solution to the problem (3). Therefore, solving (3) on active variables $\mathcal{V}$ is equivalent to restricting all inactive variables to zero. The corresponding linear program is shown in (4) in Algorithm 1. This linear program has only $|\mathcal{V}|$ variables. Now the challenge is in finding a good set of active variables $\mathcal{V}$. This set should be small, and such that the solutions of (3) and (4) are close.

We grow the set $\mathcal{V}$ greedily using the dual linear program of (4). In particular, we incrementally solve the dual by adding most violated constraints, which corresponds to adding most beneficial variables in the primal. The dual of (4) is (6) in Algorithm 2, where $\lambda \in \mathbb{R}$ is a variable associated with the distortion constraint and $\mu \in \mathbb{R}^n$ is vector of n variables associated with the simplex constraints. Given a solution (λ^*, μ^*) to the dual, the most violated constraint for a given j is the one that minimizes:

$$c_{i,j} - \lambda^* d_{i,j} - \mu_j^*. \tag{8}$$

This quantity, called the *reduced cost*, has an intuitive interpretation. We choose an example i in the direc-

Algorithm 2 DWD: Dantzig-Wolfe decomposition

```
Initialize the set of active variables:
  $\mathcal{V} \leftarrow \{(1,1), (2,2), \dots, (n,n)\}$

while the set $\mathcal{V}$ grows do
  Solve the master problem for $\lambda^*$ and $\mu^*$:
    $\underset{\lambda,\mu}{\text{maximize}} \quad \lambda\Delta + \sum_{j=1}^{n} \mu_j$    (6)
    $\text{subject to} \quad \lambda \leq 0$
    $\lambda d_{i,j} + \mu_j \leq c_{i,j} \quad \forall (i,j) \in \mathcal{V}$
  for all $j = 1, 2, \dots, n$ do
    Find the most violated constraint in the master problem for fixed $j$:
      $i^* = \arg\min_i [c_{i,j} - \lambda d_{i,j} - \mu_j]$    (7)
    if $(c_{i^*,j} - \lambda d_{i^*,j} - \mu_j < 0)$ then
      $\mathcal{V} \leftarrow \mathcal{V} \cup \{(i^*, j)\}$

Output: Active variables $\mathcal{V}$
```

tion of the steepest gradient of $f(\mathbf{X})$, so $c_{i,j}$ is small; which is close to j, so $d_{i,j}$ is close to zero (as $\lambda^* \leq 0$). The search for the most violated constraint leverages the problem structure. Therefore, our approach can be viewed as an instance of *Dantzig-Wolfe decomposition*.

The pseudocode of our search procedure is in Algorithm 2. This is an iterative algorithm, where each iteration consists of three steps. First, we solve the reduced dual linear program (6) on active variables. Second, for each point j, we identify a point i^* that minimize the reduced cost. Finally, if the pair (i^*, j) corresponds to a violated constraint, we add it to the set of active variables $\mathcal{V}$.

The pseudocode of our final solution is in Algorithm 1. We refer to Algorithm 1 as *Sparse Privacy Preserving Mappings (*SPPM*)*, because of the mappings learned by the algorithm. Algorithm 2 is a subroutine of Algorithm 1, which identifies the set of active variables $\mathcal{V}$. SPPM is parameterized by the number of iterations L.

3.3 Convergence

Algorithm SPPM is a gradient descent method. In each iteration ℓ, we find a solution $\mathbf{X}'$ in the direction of the gradient at the current solution $\mathbf{X}_{\ell-1}$. Then we find the minimum of f between $\mathbf{X}_{\ell-1}$ and $\mathbf{X}'$, and make it the next solution $\mathbf{X}_\ell$. By assumption, the initial solution $\mathbf{X}_0$ is feasible in the original problem (1). The solution $\mathbf{X}'$ to the LP (4) is always feasible in (1), because it satisfies all constraints in (1), and some additional constraints $x_{i,j} = 0$ on inactive variables. After the first iteration of SPPM, $\mathbf{X}_1$ is a convex combination of $\mathbf{X}_0$ and $\mathbf{X}'$. Since the feasible region is convex, and both $\mathbf{X}_0$ and $\mathbf{X}'$ are feasible, $\mathbf{X}_1$ is also feasible. By induction, all solutions $\mathbf{X}_\ell$ are feasible.

The value of $f(X_\ell)$ is guaranteed to monotonically decrease with ℓ. When the method converges, $f(X_\ell) = f(X_{\ell-1})$. The convergence rate of the Frank-Wolfe algorithm is $O(1/L)$ in the worst case [5].

3.4 Computational Efficiency

The computation time of our method is dominated by the search for n^2 violated constraints in Algorithm 2. To search efficiently, we implement the following speedup in the computation of the gradients $c_{i,j}$. The marginal and conditional distributions:

$$p_{\hat{B}}(\hat{b}) = \sum_{a,b} p_{A,B}(a,b) p_{\hat{B}|B}(\hat{b} \mid b)$$

$$p_{\hat{B}|A}(\hat{b} \mid a) = \frac{\sum_b p_{A,B}(a,b) p_{\hat{B}|B}(\hat{b} \mid b)}{\sum_b p_{A,B}(a,b)}$$

are precomputed, because these terms are common for all elements of $\mathbf{C}$. Then each gradient is computed as:

$$\frac{\partial}{\partial p(\hat{b}_i \mid b_j)} J(p_{a,b}, p_{\hat{b}|b}) = \sum_a p(a, b_j) \log \frac{p(\hat{b}_i \mid a)}{p(\hat{b}_i)} + \sum_a p(a, \hat{b}) \left(\frac{p(b_j \mid a)}{p(\hat{b}_i \mid a)} - \frac{p(a, b_j)}{p(\hat{b}_i)} \right).$$

Since all marginals and conditionals are precomputed, each gradient can be computed in $O(|A|)$ time.

The space complexity of our method is $O(|\mathcal{V}|)$, because we operate only on active variables $\mathcal{V}$.

We point out that the complexity of the algorithm is closely linked to the sparsity of the optimal solution, which itself is related to the value of the distortion constraint Δ. This means that some distortion regimes may not be achievable with a given computational budget. Therefore one has to reduce Δ in order to have a sparser solution. In practice however, we did not run into problems, and were able to generate mappings efficiently even when high distortion was needed to drive the mutual information close to 0.

4 Evaluation

4.1 Datasets

Census Dataset: The *Census* dataset is a sample of the United States population from 1994, and contains both categoric and numerical features. Each entry in the dataset contains features such as age, workclass, education, gender, and native country, as well as income category (smaller or larger than 50k per year). For our purposes, we consider the information to be

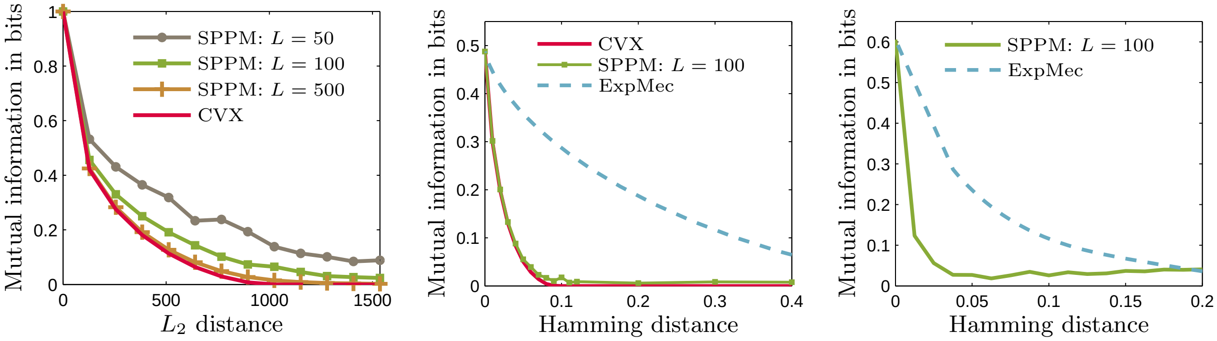

Figure 2: **Left**. Effect of parameters on privacy-distortion tradeoff. Synthetic data. **Middle**. Privacy distortion tradeoff. Census data. **Right.** Privacy distortion tradeoff. Movie data.

released publicly as the seven attributes shown in Table 1, while the income category is the private information to be protected. In this dataset, roughly 76% of the people have an income smaller than 50k. Due to the categorical nature of these features, a natural distortion metric for this data is the Hamming distance, and thus we use this metric in our experiments on this data. Fig. 3 shows that the user profile consisting of these 7 attributes can be a threat to income; the ROC curve illustrates the success rate of a simple classifier that tries to guess a user's income category when there is no privacy protection.

Movie Dataset: The well known *MovieLens* dataset [1] consists of 1M ratings of 6K users on 4K movies. Each movie comes annotated with metadata indicating its genre. In MovieLens, there are 19 genres, that we expanded as follows. We gathered the more extended set of 300 genre tags from Netflix. From these, we select those that appear in at least 5% of movies, yielding 31 genres. For user j, we compute the preference for genre i as the probability that the user chooses a movie from the genre times the reciprocal of the number of movies in that genre. We capture the user profile using a binary vector of length 31; the bits corresponding to the six most prefered genres are set to one. We treat the preference vector as public but the gender of the user as private. The fact that this profile can be a threat to gender is illustrated in Fig. 4 which shows the success of a classifier that tries to guess gender when there is no privacy protection. Once more, as the features are categorical, we use the Hamming distance for our evaluations on this dataset.

Synthetic Dataset: We consider synthetic data as well since this allows us to freely vary the problem size. The input distribution is specified in our example in Sec. 2, namely the private attribute is a binary variable $A \in \{0, 1\}$, and the public attribute B is perfectly correlated with A. By varying the parameter m as defined in the example, we modify the size of the alphabet of B, which allow us to asses the scalability. The distortion metric is the squared l_2 distance.

4.2 Benchmarks

Optimal mapping: The optimal mapping is the solution to (1); for small scale problems we were able to compute this using CVX without running out of memory. On our server, we could solve optimally (1) with alphabet size up to $|\mathcal{B}| = 2^{12} = 4096$.

Exponential Mechanism: The differential privacy metric is most commonly used in a database privacy setting, in which an analyst asks a query on a private database of size n containing data from n users. The privacy-preserving mechanism, which computes and releases the answer to the query, is designed to satisfy differential privacy under a given notion of neighboring databases. In the strong setting of *local* differential privacy [16], users do not trust the entity collecting the data in a database, thus each user holds his data locally, and passes it through a differentially private mechanism before releasing it to the untrusted entity. In this case, the privacy-preserving mechanism works on a database of size $n = 1$, and all possible databases are considered to be neighbors. This local differential privacy setting, based on input perturbation at the user end, is comparable to our local privacy setting, where user data is distorted before its release, but it differs from our setting by the privacy metric that the privacy mechanism is required to satisfy. More precisely, the local differential privacy setting considers a database of size 1 which contains the vector b of a user. The local differentially private mechanism p^{DP} satisfies $p^{DP}(\hat{b}|b) \leq e^{\varepsilon}\, p^{DP}(\hat{b}|b'), \forall b, b' \in \mathcal{B}$ and $\forall \hat{b} \in \hat{\mathcal{B}}$.

As the non-private data in our 3 datasets is categorical, we focus on the *exponential mechanism* [19], a well-known mechanism that preserves differential privacy for non-numeric valued queries. More precisely, in our experiments, we use the exponential mechanism $p^{DP}(\hat{b}|b)$ that maps b to $\hat{b}$ with a probability that decreases exponentially with the distance $d(\hat{b}, b)$

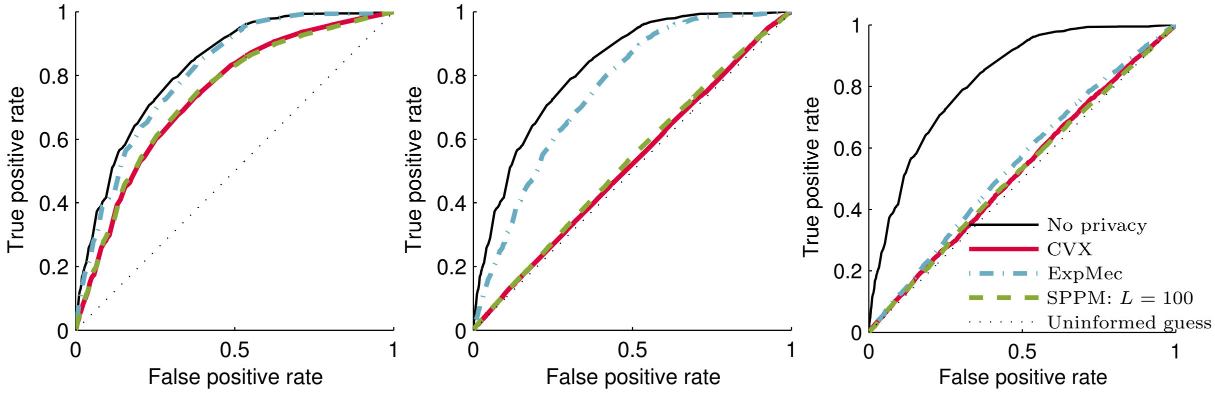

Figure 3: ROC for Naive Bayes Classifier with $\Delta = 0.02$ (Left), 0.14 (Middle) and 0.44 (Right). Census Data.

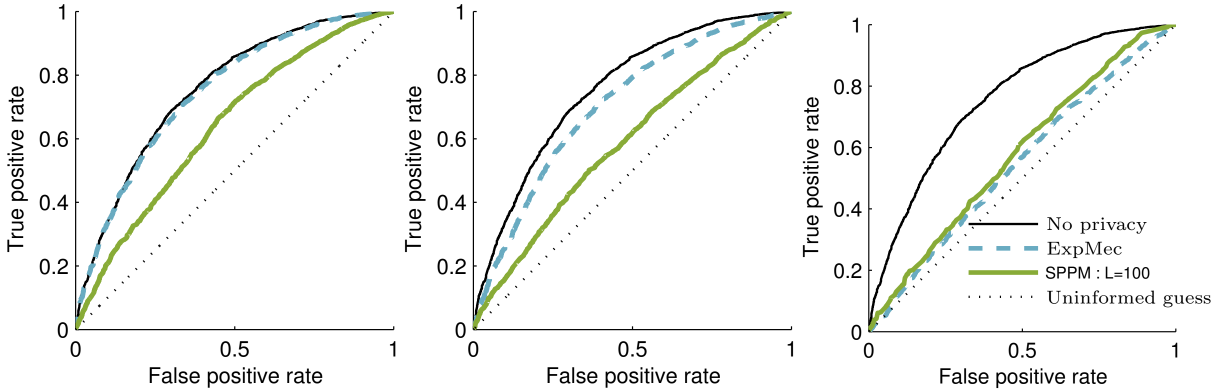

Figure 4: ROC for Logistic Regression with $\Delta = 0.04$ (Left), 0.13 (Middle) and 0.22 (Right). Movie Data.

$p^{DP}(\hat{b}|b) \propto \exp\left(-\beta d(\hat{b}, b)\right)$, where $\beta \geq 0$. Let $d_{\max} = \sup_{b,\hat{b}\in\mathcal{B}} d(\hat{b}, b)$. This exponential mechanism satisfies $(2\beta d_{max})$-local differential privacy. Intuitively, the distance $d(\hat{b}, b)$ represents how appealing substituting $\hat{b}$ for b is: the larger the distance $d(\hat{b}, b)$, the less appealing the substitution. To make a fair comparison between the exponential mechanism and SSPM, the distance $d(\hat{b}, b)$ used to define the exponential mechanism will be the same as the distance used in the distortion constraint of the optimization problem (1). In Section 4.3, $d(\hat{b}, b)$ will be the Hamming distance for experiments on the census and the movie datasets, and the squared l_2 distance for the synthetic datasets.

In [7], it was shown that in general, differential privacy with some neighboring database notion, does not guarantee low information leakage $I(A;\hat{B})$, for all priors $p_{A,B}$. However, it was also shown in [18], that *strong* ε-differential privacy, i.e. ε-differential privacy under the neighboring notion that all databases are neighbors, implies that $I(A;\hat{B}) \leq \varepsilon$. Local differential privacy is a particular case of strong differential privacy. Consequently, the mutual information between private A and the distorted $\hat{B}^{DP}$ resulting from the exponential mechanism p^{DP} will be upperbounded as $I(A;\hat{B}^{DP}) <= 2\beta d_{max}$. We acknowledge that differential privacy was not defined with the goal of minimizing mutual information. However, regardless of the mechanism, mutual information is a relevant privacy metric [7, 18, 24, 25, 22, 12], thus we can compare these 2 algorithms with respect to this metric.

4.3 Results

Parameter Choices. Using synthetic data, we explore the privacy distortion tradeoff curve (in Fig. 2 **Left**) for different values of our algorithm's parameter L. First we observe that for small values of distortions, the difference between the various curves is insignificant, which suggests that in this regime the optimal mapping is indeed sparse. Second, as we increase L accuracy improves as we approach the optimal solution. Since the gain of using $L = 500$ compared to $L = 100$ seems small, and using 100 rather than 500 approximations is clearly much faster, we elect to use $L = 100$ for further experiments.

Privacy Performance. We provide two ways of comparing SSPM and our benchmarks on two datasets. We first compare them in Fig. 2 terms of tradeoff between privacy leakage, as measured by $I(A;\hat{B})$, and

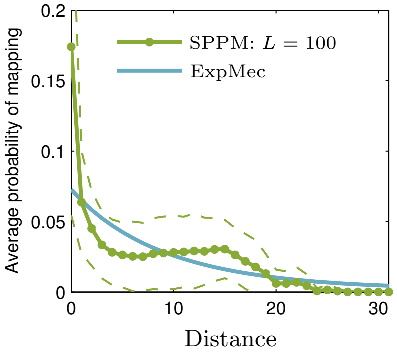

Figure 5: Behavior of SPPM and ExpMec. Synthetic data.

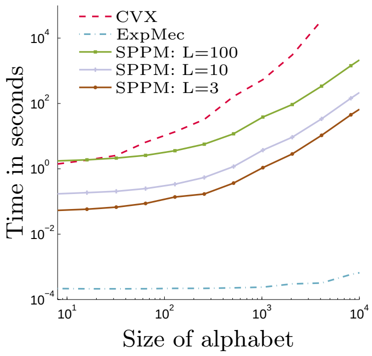

Figure 6: Time complexity and parameter sensitivity. Synthetic data.

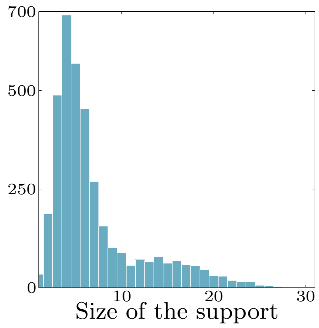

Figure 7: Histogram on size of support for mapping. Movie data, initial alphabet size 3717.

distortion. Then we draw another more general, meaningful, and fair comparison, in Fig. 3 and Fig. 4, by comparing the ability of SSPM and our benchmarks to defeat two different inference algorithms that would try to infer private data by using the privatized data $\hat{B}$. ROC curves are agnostic and meaningful measures privacy that do not a priori favor a privacy metric (differential privacy or mutual information).

We start by illustrating the privacy versus distortion tradeoff for SPPM and our benchmarks, on the Census dataset, in which each user is represented by a vector of 7 attributes. We do not consider all possible values of this vector, as it would be prohibitive for an exact solver and prevent us from comparing to an optimal solution. Instead, we restrict the alphabet $|\mathcal{B}|$ to the 300 most probable vectors of 7 attributes, since we are mainly focused on a relative comparison. In Fig. 2(**Middle**) we see that SPPM is nearly indistinguishable from the optimal solution whereas the exponential mechanism (ExpMec) is much further away. To bring the mutual information down from 0.5 to 0.07, SPPM needs 0.05 distortion to achieve perfect privacy, while ExpMec needs more than 8 times as much. Note that for a given level of distortion, e.g. 0.1, 0.2, SPPM achieves much better privacy than ExpMec as the mutual information is significantly lower.

Next we consider the Movie dataset which is one order of magnitude larger than the Census dataset. The results in Fig. 2(**Right**) mirror what we observed with the Census data, namely that a given level of privacy can be achieved with less distortion using SPPM as opposed to ExpMec. For example, to reduce our privacy leakage metric from 0.6 to 0.05, SPPM requires roughly 0.03 distortion whereas ExpMec needs approximately 0.17, nearly 6 times as much. For both the Census and Movie datasets, SPPM can achieve perfect or near-perfect privacy with a small distortion.

Differential privacy does not aim to minimize mutual information, which explains why ExpMec does not perform as well as SSPM in Fig. 2. Another metric to gage the success of our privacy mapping is to consider its impact on a classifier attempting to infer the private attribute. Recall that the goal of our mapping is to weaken any classifier that threatens to infer the private attribute. First, we consider a simple Naive Bayes classifier that analyzes the Census data to infer each user's income category. We quantify the classifier's success, in terms of true positives and false negatives (in an ROC curve) in Fig. 3. Recall that in an ROC curve, the $y = x$ line corresponds to a blind classifier that is no better than an uninformed guess. The weaker a classifier the closer it is to this line, and it becomes useless when it matches this line. We consider three bounds on distortion that allow us to explore the extremes of nearly no distortion ($\Delta = 0.02$ in Fig. 3(a)), a large amount of distortion ($\Delta = 0.44$ in Fig. 3(c)), and something in between ($\Delta = 0.14$ in Fig. 3(b)). In the case of small distortion, all algorithms make modest improvements over the no privacy case. However even in this scenario, SPPM performs close to optimal, unlike ExpMec that only slightly outperforms the no privacy case. With only a very small amount of distortion, not even the optimal solution can render the classifier completely useless (i.e. equivalent to the blind uninformed guess). On the other hand, when a large distortion is permitted, then all algorithms do naturally well (Fig. 3(c)). For a value of Δ between these extremes, SPPM is close to optimal, while ExpMec can only weaken the classifier a little.

In a second scenario, we study a logistic regression classifier that analyzes the movie dataset to infer gender. We focus on logistic regression for movie data because it has been shown to be an effective classifier for inferring gender [27]. Again we see in Fig. 4 that the findings essentially mimic those of the previous case. We thus conclude that, for a given distortion budget, SPPM is more successful against inference threats because it can diminish the success of a classifier more than the exponential mechanism. In other words, SPPM can provide more privacy than the exponential mechanism for the same level of distortion.

Why is it that SPPM consistently outperforms ExpMec? Fig. 5 illustrates how these mappings work. We plot the average probability of being mapped to the kth closest point, together with the standard error that measures the variability among points. In ExpMec, the probability of mapping one point to another decreases exponentially with distance and the same mapping is applied to all points (in other words the standard deviation is null). With SPPM, many points are mapped to themselves. This means that the privacy of these points cannot be improved within the distortion constraint. This can happen if a given profile is already very private, or if there is no hope of providing privacy. In both cases, the distortion budget should not be wasted on such profiles, and SPPM is able to detect these cases and save its budget for other cases. ExpMec *wastes* some distortion on those points which are forced to be mapped to close neighbours. The key difference between SPPM and ExpMec is that SPPM is not required to apply the same mapping to each user, and can thus personalize the distortion, or adapt it as needed. This flexibility leads to a fair bit of variance as seen in Fig. 5. Because the probability of mapping to another point does not decrease with the distance from that point, we see a non-monotonically decreasing curve with distance: some points are better off being mapped to far neighbors rather than close ones; e.g. users whose profile is hard to disguise.

To further understand the effect of the mappings that SPPM proposes, we examine which features in a user profile are impacted most by our distortions. In Tab. 1 we list the mutual information between a single public feature (e.g., Education) and the private attribute we wish to hide (e.g., income category). We see that $I(A;F)$ is largest for Education, Marital status and Occupation, indicating these features are the most correlated with the private attribute. This is intuitive as, for example, more highly educated people tend to make larger salaries. The table shows that these 3 features experience the largest reduction in mutual information after distortion, indicating that SPPM spent its distortion budget on the biggest threats. This intuitive property shows that the mappings learned depend on the underlying prior distribution in a *smart* way, such that with limited distortion budget the priority is on the biggest privacy threats. Another point can be easily seen in the movie data in Tab. **??**; here the single features mutual information are rather low, whereas the mutual information of the full vector of features is significant. This means that a big part of the privacy threat comes from co-occurrence of features rather than individual features. This explains why it is not enough to simply reduce the mutual information of a single feature to obtain good privacy.

		$I(A;F)$	
Feature F	Examples	before	after
Age	10-20, 20-30,...	0.1064	0.0408
Education	Bachelor, PhD,...	0.1502	0.0702
Marital status	Divorced, Married,...	0.1241	0.0440
Occupation	Manager, Scientist,...	0.1126	0.0367
Race	Black, White,...	0.0192	0.0084
Gender	Female, Male	0.0123	0.0082
Country	Mexico, USA,...	0.0470	0.0203

Table 1: Mutual information between private attribute A and public attributes F before and after SPPM on Census dataset.

Scalability. Having established good privacy performance, we now assess how the runtime performance scales in terms of the size of the problem, shown in Fig. 6. We fix the distortion constraint to be proportional to the size of the problem, in order to keep a similar difficulty as we grow the size. As stated in Sec. 3.1, the time complexity is linear in L. This trend is evident as we observe the gaps between the lines for $L = 3, 10$, and 100. Importantly, we see that our method scales with problem size better than the optimal solution. We observe that the computation speed of the exponential mechanism is very quick, and note that indeed this is one of the salient properties of this mechanism. Overall, this figure shows that our method is indeed tractable and can compute the distortion maps within a few minutes for problems whose alphabet size is on the order of tens of thousands.

Recall that we designed for computation efficiency by smartly selecting a limited number of alternate user profiles to map each original profile to. This corresponds to the support size of $p(\hat{B}|b_i)$ for all b_i. In Fig. 7 we show the histogram of the support sizes for our SPPM mapping on the movie data. Although the initial alphabet size is very large (above 3500), for most users we don't consider more than 10 alternate profiles, and in the worst case we consider up to 30. This amounts to huge savings in computation and memory, without sacrificing privacy.

5 Conclusion

In this paper, for the first time, we apply large scale LP optimization techniques to the problem of data distortion for privacy. We show that our privacy-preserving mappings can be close to optimal, and consistently outperform a state of the art technique called the Exponential Mechanism. Our solution achieves better privacy with less distortion than existing solutions, when privacy leakage is measured by a mutual information metric. We demonstrated that our method can scale, even for systems with many users and a large underlying alphabet that describes their profiles.

References

[1] MovieLens Dataset, GroupLens Research. `http://www.grouplens.org/datasets/movielens/`, 2010.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. *ACM Sigmod Record*, 2000.

[3] B. Fung and K. Wang and R. Chen and P. Yu. Privacy Preserving Data Publishing: A Survey of Recent Developments. *ACM Computing Surveys*, 2010.

[4] Siddhartha Banerjee, Nidhi Hegde, and Laurent Massoulié. The price of privacy in untrusted recommendation engines. In *IEEE Allerton*, 2012.

[5] Dimitri Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1999.

[6] C. Jernigan and B. Mistree. Gaydar: Facebook Friendships expose sexual orientation. In *First Monday*, October 2009.

[7] Flavio Calmon and Nadia Fawaz. Privacy against statistical inference. In *IEEE Allerton*, 2012.

[8] Kenneth L. Clarkson. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Trans. Algorithms*, September 2010.

[9] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy and statistical minimax rates. In *IEEE FOCS*, 2013.

[10] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[11] Cynthia Dwork. Differential privacy. In *Automata, Languages and Programming*. Springer, 2006.

[12] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *ACM PODS*, 2003.

[13] Martin Jaggi. *Sparse Convex Optimization Methods for Machine Learning*. PhD thesis, 2011.

[14] Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. *submitted*, 2012.

[15] Lee K. Jones. A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural network training. *The Annals of Statistics*, 1992.

[16] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Jour. on Computing*, 2011.

[17] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM TKDD*, 2007.

[18] Ali Makhdoumi and Nadia Fawaz. Privacy-utility tradeoff under statistical uncertainty. In *IEEE Allerton*, 2013.

[19] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, 2007.

[20] Nina Mishra and Mark Sandler. Privacy via pseudorandom sketches. In *ACM PODS*, 2006.

[21] J. Otterbacher. Inferring gender of movie reviewers: exploiting writing style, content and metadata. In *CIKM*, 2010.

[22] D. Rebollo-Monedero, J. Forné, and J. Domingo-Ferrer. From t-closeness-like privacy to postrandomization via information theory. *IEEE Trans. on Knowledge and Data Engineering*, 2010.

[23] I. S. Reed. Information Theory and Privacy in Data Banks. In *AFIPS '73*, pages 581–587, 1973.

[24] Salman Salamatian, Amy Zhang, Flavio du Pin Calmon, Sandilya Bhamidipati, Nadia Fawaz, Branislav Kveton, Pedro Oliveira, and Nina Taft. How to hide the elephant- or the donkey- in the room: Practical privacy against statistical inference for large data. In *IEEE GlobalSIP*, 2013.

[25] L. Sankar, S. R. Rajagopalan, and H. V. Poor. Utility-privacy tradeoff in databases: An information-theoretic approach. *IEEE Trans. Inf. Forensics Security*, 2013.

[26] Latanya Sweeney. k-anonymity: A model for protecting privacy. *Internat. Jour. Uncertainty, Fuzziness and Knowledge-Based Systems*, 2002.

[27] U. Weinsberg and S. Bhagat and S. Ioannidis and N. Taft. BlurMe: Inferring and Obfuscating User Gender Based on Ratings. In *ACM RecSys*, 2012.

[28] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Jour. American Statistical Association*, 1965.

[29] H. Yamamoto. A source coding problem for sources with additional outputs to keep secret from the receiver of wiretappers. *IEEE Trans. Information Theory*, 29(6), 1983.

[30] Tong Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Trans. Inf. Theor.*, 49(3):682–691, 2006.

[31] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003.