Differentially Private Database Release via Kernel Mean Embeddings

Matej Balog [1 2] **Ilya Tolstikhin** [1] **Bernhard Schölkopf** [1]

Abstract

We lay theoretical foundations for new database release mechanisms that allow third-parties to construct consistent estimators of population statistics, while ensuring that the privacy of each individual contributing to the database is protected. The proposed framework rests on two main ideas. First, releasing (an estimate of) the kernel mean embedding of the data generating random variable instead of the database itself still allows third-parties to construct consistent estimators of a wide class of population statistics. Second, the algorithm can satisfy the definition of differential privacy by basing the released kernel mean embedding on entirely synthetic data points, while controlling accuracy through the metric available in a Reproducing Kernel Hilbert Space. We describe two instantiations of the proposed framework, suitable under different scenarios, and prove theoretical results guaranteeing differential privacy of the resulting algorithms and the consistency of estimators constructed from their outputs.

1. Introduction

We aim to contribute to the body of research on the trade-off between releasing datasets from which publicly beneficial statistical inferences can be drawn, and between protecting the privacy of individuals who contribute to such datasets. Currently the most successful formalisation of protecting user privacy is provided by *differential privacy* (Dwork & Roth, 2014), which is a *definition* that any algorithm operating on a database may or may not satisfy. An algorithm that does satisfy the definition ensures that a particular individual does not lose too much privacy by deciding to contribute to the database on which the algorithm operates.

While differentially private algorithms for releasing entire databases have been studied previously (Blum et al., 2008; Wasserman & Zhou, 2010; Zhou et al., 2009), most algorithms focus on releasing a privacy-protected version of a particular summary statistic, or of a statistical model trained on the private dataset. In this work we revisit the more difficult *non-interactive*, or *offline* setting, where the database owner aims to release a privacy-protected version of the entire database without knowing what statistics third-parties may wish to compute in the future.

In our new framework we propose to use the kernel mean embedding (Smola et al., 2007) as an intermediate representation of a database. It is (1) sufficiently rich in the sense that it captures a wide class of statistical properties of the data, while at the same time (2) it lives in a Reproducing Kernel Hilbert Space (RKHS), where it can be handled mathematically in a principled way and privacy-protected in a unified manner, independently of the type of data appearing in the database. Although kernel mean embeddings are functions in an abstract Hilbert space, in practice they can be (at least approximately) represented using a possibly weighted set of data points in input space (i.e. a set of database rows). The privacy-protected kernel mean embedding is released to the public in this representation, however, using synthetic datapoints instead of the private ones. As a result, our framework can be seen as leading to *synthetic database* algorithms.

We validate our approach by instantiating two concrete algorithms and proving that they output consistent estimators of the true kernel mean embedding of the data generating process, while satisfying the definition of differential privacy. The consistency results ensure that third-parties can carry out a wide variety of statistically founded computation on the released data, such as constructing consistent estimators of population statistics, estimating the Maximum Mean Discrepancy (MMD) between distributions, and two-sample testing (Gretton et al., 2012), or using the data in the kernel probabilistic programming framework for random variable arithmetics (Schölkopf et al., 2015; Simon-Gabriel et al., 2016, Section 3), repeatedly and unlimitedly without being able to, or having to worry about, violating user privacy.

One of our algorithms is especially suited to the interesting scenario where a (small) subset of a database has already been published. This situation can arise in a wide variety of settings, for example, due to weaker privacy protections in

[1]MPI-IS, Tübingen, Germany [2]University of Cambridge, UK. Correspondence to: Matej Balog <first.surname@gmail.com>. Code: https://github.com/matejbalog/RKHS-private-database/.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

the past, due to a leak, or due to the presence of an incentive, financial or otherwise, for users to publish their data. In such a situation our algorithm provides a principled approach for reweighting the public data in such a way that the accuracy of statistical inferences on this dataset benefits from the larger sample size (including the private data), while maintaining differential privacy for the undisclosed data.

In summary, the contributions of this paper are:

1. A new framework for designing database release algorithms with the guarantee of differential privacy. The framework uses kernel mean embeddings as intermediate database representations, so that the RKHS metric can be used to control accuracy of the released synthetic database in a principled manner (Section 3).
2. Two instantiations of our framework in the form of two synthetic database algorithms, with proofs of their consistency, convergence rates and differential privacy, as well as basic empirical illustrations of their performance on synthetic datasets (Sections 4 and 5).

2. Background

2.1. Differential Privacy

Definition 1 (Dwork, 2006)**.** For $\varepsilon > 0, \delta \geq 0$, algorithm $\mathcal{A}$ is said to be (ε, δ)-differentially private if for all neighbouring databases $D \sim D'$ (differing in at most one element) and all measurable subsets S of the co-domain of $\mathcal{A}$,

$$\mathbb{P}\left(\mathcal{A}(D) \in S\right) \leq e^{\varepsilon}\mathbb{P}\left(\mathcal{A}(D') \in S\right) + \delta. \tag{1}$$

The parameter ε controls the amount of information the algorithm can leak about an individual, while a positive δ allows the algorithm to produce an unlikely output that leaks more information, but only with probability up to δ. This notion is sometimes called *approximate* differential privacy; an algorithm that is $(\varepsilon, 0)$-differentially private is simply said to be ε-differentially private. Note that any non-trivial differentially private algorithm must be randomised; the definition asserts that the distribution of algorithm outputs is not too sensitive to changing one row in the database.

When the algorithm's output is a finite vector $\mathcal{A}(D) \in \mathbb{R}^J$, two standard random perturbation mechanisms for making this output differentially private are the *Laplace* and *Gaussian* mechanisms. As the perturbation needs to mask the contribution of each individual entry of the database D, the scale of the added noise is closely linked to the notion of *sensitivity*, measuring how much the algorithm's output can change due to changing a single data point:

$$\Delta_1 := \sup_{D \sim D'} \left\|\mathcal{A}(D) - \mathcal{A}(D')\right\|_1, \tag{2}$$

$$\Delta_2 := \sup_{D \sim D'} \left\|\mathcal{A}(D) - \mathcal{A}(D')\right\|_2. \tag{3}$$

The Laplace mechanism adds i.i.d. $\mathrm{Lap}(\Delta_1/\varepsilon)$ noise to each of the J coordinates of the output vector and ensures pure ε-differential privacy, while the Gaussian mechanism adds i.i.d. $\mathcal{N}(0, \sigma^2)$ noise to each coordinate, where $\sigma^2 > 2\Delta_2^2 \ln(1.25/\delta)/\varepsilon^2$, and ensures (ε, δ)-differential privacy. Applying these mechanisms thus requires computing (an upper bound on) the relevant sensitivity.

Differential privacy is preserved under post-processing: if an algorithm $\mathcal{A}$ is (ε, δ)-differentially private, then so is its sequential composition $\mathcal{B}(\mathcal{A}(\cdot))$ with any other algorithm $\mathcal{B}$ that does not have direct or indirect access to the private database D (Dwork & Roth, 2014).

2.2. Kernels, RKHS, and Kernel Mean Embeddings

A kernel on a non-empty set (data type) $\mathcal{X}$ is a binary positive-definite function $k(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Intuitively it can be thought of as expressing the similarity between any two elements in $\mathcal{X}$. The literature on kernels is vast and their properties are well studied (Schölkopf & Smola, 2002); many kernels are known for a large variety of data types such as vectors, strings, time series, graphs, etc, and kernels can be composed to yield valid kernels for composite data types (e.g. the type of a database row containing both numerical and string data).

The *kernel mean embedding* (KME) of an $\mathcal{X}$-valued random variable X in the RKHS is the function $\mu_X^k : \mathcal{X} \to \mathbb{R}$, $y \mapsto \mathbb{E}_X[k(X, y)]$, defined whenever $E_X[\sqrt{k(X, X)}] < \infty$ (Smola et al., 2007). Several popular kernels have been proved to be *characteristic* (Fukumizu et al., 2008), in which case the map $p_X \mapsto \mu_X^k$, where p_X is the distribution of X, is injective. This means that no information about the distribution of X is lost when passing to its KME μ_X^k.

In practice, the KME of a random variable X is approximated using a sample $x_1, \ldots, x_N$ drawn from X, which can be used to construct an *empirical KME* $\hat{\mu}_X^k$ of X in the RKHS: a function given by $y \mapsto \frac{1}{N}\sum_{n=1}^{N} k(x_n, y)$. When the x_n's are i.i.d., under a boundedness condition $\hat{\mu}_X^k$ converges to the true KME μ_X^k at rate $\mathcal{O}_p(N^{-1/2})$, independently of the dimension of $\mathcal{X}$ (Lopez-Paz et al., 2015)[1]. Our approach relies on the metric of the RKHS in which these KMEs live. The RKHS $\mathcal{H}_k$ is a space of functions, endowed with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$ that satisfies the *reproducing* property $\langle k(x, \cdot), h \rangle = h(x)$ for all $x \in \mathcal{X}$ and $h \in \mathcal{H}_k$. The inner product induces a norm $\| \cdot \|_{\mathcal{H}_k}$, which can be used to measure distances $\|\mu_X^k - \mu_Y^k\|_{\mathcal{H}_k}$ between distributions of X and Y. This can be exploited for various

[1] The KME can be viewed as a smoothed version of the density, which is easier to estimate than the density itself; rates of nonparametric density estimation or statistical powers of two-sample or independence tests involving p_X are known to necessarily degrade with growing dimension (Tolstikhin et al., 2017, Section 4.3).

purposes such as two-sample tests (Gretton et al., 2012), independence testing (Gretton et al., 2005), or one can attempt to minimise this distance in order to match one distribution to another.

An example of such minimisation are *reduced set methods* (Burges, 1996; Schölkopf & Smola, 2002, Chap. 18), which replace a set of points $S = \{x_1, \dots, x_N\} \subseteq \mathcal{X}$ with a weighted set $R = \{(z_1, w_1), \dots, (z_M, w_M)\} \subseteq \mathcal{X} \times \mathbb{R}$ (of potentially smaller size), where the new points z_m can, but need not equal any of the x_ns, such that the KME computed using the reduced set R is close to the KME computed using the original set S, as measured by the RKHS norm:

$$\left\|\mu_S^k - \mu_R^k\right\|_{\mathcal{H}_k} = \left\| \frac{1}{N}\sum_{n=1}^{N} k(x_n, \cdot) - \sum_{m=1}^{M} w_m k(z_m, \cdot) \right\|_{\mathcal{H}_k}.$$

Reduced set methods are usually motivated by the computational savings arising when $|R| < |S|$; we will invoke them mainly to replace a collection S of private data points with a (possibly weighted) set R of synthetic data points.

3. Framework

3.1. Problem Formulation

Throughout this work, we assume the following setup. A database curator wishes to publicly release a database $D = \{x_1, \dots x_N\} \in \mathcal{X}^N$ containing private data about N individuals, with each data point (database row) x_n taking values in a non-empty set $\mathcal{X}$. The set $\mathcal{X}$ can be arbitrarily rich, for example, it could be a product of Euclidean spaces, integer spaces, sets of strings, etc.; we only require availability of a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ on $\mathcal{X}$. We assume that the N rows $x_1, \dots, x_N$ in the database D can be thought of as i.i.d. observations from some $\mathcal{X}$-valued data-generating random variable X (but see Section 7 for a discussion about relaxing this assumption). The database curator, wishing to protect the privacy of individuals in the database, seeks a database release mechanism that satisfies the definition of (ε, δ)-differential privacy, with $\varepsilon > 0$ and $\delta \geq 0$ given. The main purpose of releasing the database is to allow third parties to construct estimators of population statistics (i.e. properties of the distribution of X), but it is not known at the time of release what statistics the third-parties will be interested in.

To lighten notation, henceforth we drop the superscript k from KMEs (such as μ_X^k) and the subscript k from the RKHS $\mathcal{H}_k$, whenever k is the kernel on $\mathcal{X}$ chosen by the database curator.

3.2. Algorithm Template

We propose the following general algorithm template for differentially private database release:

1. Construct a consistent estimator $\hat{\mu}_X$ of the KME μ_X of X using the private database.

2. Obtain a perturbed version $\tilde{\mu}_X$ of the constructed estimate $\hat{\mu}_X$ to ensure differential privacy.

3. Release a (potentially approximate) representation of $\tilde{\mu}_X$ in terms of a (possibly weighted) dataset $\{(z_1, w_1), \dots, (z_M, w_M)\} \subseteq \mathcal{X} \times \mathbb{R}$.

The released representation should be such that $\sum_{m=1}^{M} w_m k(z_m, \cdot)$ is a consistent estimator of the true KME μ_X, i.e. such that the RKHS distance between the two converges to 0 in probability as the private database size N, and together with it the synthetic database size M, go to infinity.

Each step of this template admits several possibilities. For the first step we have discussed the standard empirical KME $\frac{1}{N}\sum_{n=1}^{N} k(x_n, \cdot)$ with $x_1, \dots, x_N$ i.i.d. observations of X, but the framework remains valid with improved estimators such as *kernel-based quadrature* (Chen et al., 2010) or the *shrinkage* kernel mean estimators of (Muandet et al., 2016).

As the KMEs $\hat{\mu}_X$ and μ_X live in the RKHS $\mathcal{H}$ of the kernel k, a natural mechanism for privatising $\hat{\mu}_X$ in the second step would be to follow (Hall et al., 2013) and pointwise add to $\hat{\mu}_X$ a suitably scaled sample path g of a Gaussian process with covariance function $k(\cdot, \cdot)$. This does ensure (ε, δ)-differential privacy of the resulting function $\tilde{\mu}_X = \hat{\mu}_X + g$, but unfortunately $\tilde{\mu}_X \notin \mathcal{H}$, because the RKHS norm $\|g\|_{\mathcal{H}}$ of a Gaussian process sample path with the same kernel k is infinite almost surely (Rasmussen & Williams, 2005). While our framework allows pursuing this direction by, for example, moving to a larger function space that does contain the Gaussian process sample path, in this work we will present algorithms that achieve differential privacy by mapping $\hat{\mu}_X$ into a finite-dimensional Hilbert space and then employing the standard Laplace or Gaussian mechanisms to the finite coordinate vector.

Differential privacy is preserved under post-processing, but the third step does require some care to ensure that private data is not leaked. Specifically, when several possible (approximate) representations $\tilde{\mu}_X \approx \sum_{m=1}^{M} w_m k(z_m, \cdot)$ in terms of a weighted dataset $(w_1, z_1), \dots, (w_M, z_M)$ are possible, committing to a particular one reveals more information than just the function $\tilde{\mu}_X$ (consider, for example, the extreme case where the representation would be in terms of the private points $x_1, \dots, x_N$). One thus needs to either control the privacy leak due to choosing a representation in a way that depends on the private data, or, as we do in our concrete algorithms below, choose a representation independently of the private data (but still minimising its RKHS distance to the privacy-protected $\tilde{\mu}_X$).

3.3. Versatility

Algorithms in our framework release a possibly weighted synthetic dataset $\{(z_1, w_1), \ldots, (z_M, w_M)\} \subseteq \mathcal{X} \times \mathbb{R}$ such that $\sum_{m=1}^{M} w_m k(z_m, \cdot)$ is a consistent estimator of the true KME μ_X of the data generating random variable X. This allows third-parties to perform a wide spectrum of statistical computation, all without having to worry about violating differential privacy:

1. *Kernel probabilistic programming* (Schölkopf et al., 2015): The versatility of our approach is greatly expanded thanks to the result of (Simon-Gabriel et al., 2016), who showed that under technical conditions, applying a continuous function f to all points z_m in the synthetic dataset yields a consistent estimator $\sum_{m=1}^{M} w_m k_f(f(z_m), \cdot)$ of the KME $\mu_{f(X)}$ of the transformed random variable $f(X)$, even when the points $z_1, \ldots, z_M$ are not i.i.d. (as they may not be, depending on the particular synthetic database release algorithm).

2. *Consistent estimation of population statistics*: For any RKHS function $h \in \mathcal{H}$, we have $\langle \mu_X, h \rangle_{\mathcal{H}} = \mathbb{E}[h(X)]$, so a consistent estimator of μ_X yields a consistent estimator of the expectation of $h(X)$. It can be evaluated using the reproducing kernel property:

$$\mathbb{E}[h(X)] = \langle \mu_X, h \rangle_{\mathcal{H}} \approx \left\langle \sum_{m=1}^{M} w_m k(z_m, \cdot), h \right\rangle_{\mathcal{H}} = \sum_{m=1}^{M} w_m h(z_m). \tag{4}$$

   For example, approximating the indicator function $\mathbb{1}_S$ of a set $S \subseteq \mathcal{X}$ with functions in the RKHS allows estimating probabilities: $\mathbb{E}[\mathbb{1}_S(X)] = \mathbb{P}[X \in S]$ (note that $\mathbb{1}_S$ itself may not be an element of the RKHS).

3. *MMD estimation and two-sample testing* (Gretton et al., 2012): Given another random variable Y on $\mathcal{X}$, one can consistently estimate the Maximum Mean Discrepancy (MMD) distance $\|\mu_X - \mu_Y\|_{\mathcal{H}}$ between the distributions of X and Y, and in particular to construct a two-sample test based on this distance. Given a sample $y_1, \ldots, y_L \sim Y$:

$$\|\mu_X - \mu_Y\|_{\mathcal{H}} \approx \left\| \sum_{m=1}^{M} w_m k(z_m, \cdot) - \frac{1}{L} \sum_{l=1}^{L} k(y_l, \cdot) \right\|_{\mathcal{H}},$$

   which can again be evaluated using the reproducing property.

4. *Subsequent use of synthetic data*: Since the output of the algorithm is a (possibly weighted) database, third-parties are free to use this data for arbitrary purposes, such as training any machine learning model on this data. Models trained purely on this data can be released with differential privacy guaranteed; however, the accuracy of such models on real data remains an empirical question that is beyond the scope of this work.

An orthogonal spectrum of versatility arises from the fact that the third step in the algorithm template can constrain the released dataset $(z_1, w_1), \ldots, (z_M, w_M)$ to be more convenient or more computationally efficient for further processing. For example, one could fix the weights to uniform $w_m = \frac{1}{M}$ to obtain an unweighted dataset, or to replace an expensive data type with a cheaper subset, such as requesting floats instead of doubles in the z_m's. All this can be performed while an RKHS distance is available to control accuracy between $\tilde{\mu}_X$ and its released representation.

3.4. Concrete Algorithms

As a first illustrative example, we describe how a particular case of an existing, but inefficient synthetic database algorithm already fits into our framework. The *exponential mechanism* (McSherry & Talwar, 2007) is a general mechanism for ensuring ε-differential privacy, and in our setting it operates as follows: given a similarity measure $s : \mathcal{X}^N \times \mathcal{X}^M \to \mathbb{R}$ between (private) databases of size N and (synthetic) databases of size M, output a random (synthetic) database R with probability proportional to $\exp(\frac{\varepsilon}{2\Delta_1} s(D, R))$, where D is the actual private database and Δ_1 is the L_1 sensitivity of s w.r.t. D. This ensures ε-differential privacy (McSherry & Talwar, 2007).

To fit this into our framework, we can take $s(D, R) = -\|\mu_D - \mu_R\|_{\mathcal{H}}$ to be the negative RKHS distance between the KMEs computed using D and R, and achieve ε-differential privacy by releasing R with probability proportional to $\exp(-\frac{\varepsilon}{2\Delta_1}\|\mu_D - \mu_R\|_{\mathcal{H}})$. This solves steps 2 and 3 of our general algorithm template simultaneously, as it directly samples a concrete representation of a "perturbed" KME μ_R. The algorithm essentially corresponds to the SmallDB algorithm of Blum et al. (2008), except for choosing the RKHS distance as a well-studied similarity measure between two databases.

The principal issue with this algorithm is its computational infeasibility except in trivial cases, as it requires sampling from a probability distribution supported on all potential synthetic databases, and employing an approximate sampling scheme can break the differential privacy guarantee of the exponential mechanism. In Sections 4 and 5 respectively, we describe two concrete synthetic database release algorithms that may possess failure modes where they become inefficient, but employing approximations in those cases can only affect their statistical accuracy, not the promise of differential privacy.

Algorithm 1 Differentially private database release via a synthetic data subspace of the RKHS

Input: database $D = \{x_1, \dots, x_N\} \subseteq \mathcal{X}$, kernel k on $\mathcal{X}$, privacy parameters $\varepsilon > 0$ and $\delta > 0$
Output: (ε, δ)-differentially private, weighted synthetic database (representing an estimate of μ_X in the RKHS $\mathcal{H}$ of k)
1: $M \leftarrow M(N) \in \omega(1) \cap o(N^2)$, number of synthetic data points to use
2: $z_1, \dots, z_M \leftarrow$ initialised deterministically or randomly from some distribution q on $\mathcal{X}$
3: $\mathcal{H}_M \leftarrow \mathrm{Span}(\{k(z_1, \cdot), \dots, k(z_M, \cdot)\}) \leq \mathcal{H}$
4: $b_1, \dots, b_F \leftarrow$ orthonormal basis of $\mathcal{H}_M$ (obtained using, e.g. Gram-Schmidt)
5: $\hat{\mu}_X \leftarrow \frac{1}{N} \sum_{n=1}^{N} k(x_n, \cdot)$, empirical KME of X in $\mathcal{H}$
6: $\overline{\mu}_X \leftarrow \sum_{f=1}^{F} \langle b_f, \hat{\mu}_X \rangle_{\mathcal{H}} b_f =: \sum_{f=1}^{F} \alpha_f b_f$, projection of $\hat{\mu}_X$ onto $\mathcal{H}_M$
7: $\boldsymbol{\beta} \leftarrow \boldsymbol{\alpha} + \mathcal{N}(0, \frac{8 \ln(1.25/\delta)}{N^2 \varepsilon^2} I_F)$, an (ε, δ)-differentially private version of the coordinate vector $\boldsymbol{\alpha}$ (Gaussian mechanism)
8: $\tilde{\mu}_X \leftarrow \sum_{f=1}^{F} \beta_f b_f = \sum_{m=1}^{M} w_m k(z_m, \cdot)$, re-expressed in terms of $k(z_m, \cdot)$'s
9: **return** $(z_1, w_1), \dots, (z_M, w_M)$

4. Perturbation in Synthetic-Data Subspace

In this section we describe an instantiation of the framework proposed in Section 3 that achieves differential privacy of the KME by projecting it onto a finite-dimensional subspace of the RKHS spanned by feature maps $k(z_m, \cdot)$ of synthetic data points $z_1, \dots, z_M$, and perturbing the resulting finite coordinate vector. To ensure differential privacy, the synthetic data points are chosen independently of the private database. As a result, statistical efficiency of this approach will depend on the choice of synthetic data points, with efficiency increasing if there are enough synthetic data points to capture the patterns in the private data. Therefore this algorithm is especially suited to the scenario discussed in Section 1, where a part of the database (or of a similar one) has already been published, as this can serve as a good starting set for the synthetic data points.

The setting where some observations from X have already been released highlights the fact that differential privacy only protects against *additional* privacy violation due to an individual deciding to contribute to the private database; if a particular user's data has already been published, differential privacy does not protect against privacy violations based on exploiting this previously published data.

The algorithm is formalised as Algorithm 1 above. Lines 1-2 choose synthetic data points $z_1, \dots, z_M$ independently of the private data (only using the database size N). Lines 3-4 construct the linear subspace $\mathcal{H}_M$ of $\mathcal{H}$ spanned by feature maps of the chosen synthetic data points, and compute a (finite) basis for it. Only then the private data is accessed: the empirical KME $\hat{\mu}_X$ is computed (line 5), projected onto the subspace $\mathcal{H}_M$ and expressed in terms of the precomputed basis (line 6). The basis coefficients of the projection are then perturbed to achieve differential privacy (line 7), and the perturbed element $\tilde{\mu}_X \in \mathcal{H}_M$ is then re-expressed in terms of the spanning set containing feature maps of synthetic data points (line 8). This expansion is finally released to the public (line 9).

Line 1 stipulates that the number of synthetic data points $M \to \infty$ as $N \to \infty$, but asymptotically slower than N^2. This is to ensure that the privatisation noise added in the subspace $\mathcal{H}_M$ to each coordinate is small enough overall to preserve consistency, as stated in the following Theorem 2. This theorem assures us that Algorithm 1 produces a consistent estimator of the true KME μ_X, if the synthetic data points are sampled from a distribution with sufficiently large support. Due to space constraints, all proofs appear in Appendix A.

Theorem 2. *Let $\mathcal{X}$ be a compact metric space and k a continuous kernel on $\mathcal{X}$. If the synthetic data points $z_1, z_2, \dots$ are sampled i.i.d. from a distribution q on $\mathcal{X}$ such that the support of X is included in the support of q, then Algorithm 1 outputs a consistent estimator of the KME μ_X: $\sum_{m=1}^{M} w_m k(z_m, \cdot) \xrightarrow{\mathbb{P}} \mu_X$ as $N \to \infty$.*

As discussed by Simon-Gabriel et al. (2016), these assumptions are usually satisfied: $\mathcal{X}$ can be taken to be compact whenever the data comes from measurements with any bounded range, and many kernels are continuous, including all kernels on discrete spaces (w.r.t. to the discrete topology).

In order to use the output of Algorithm 1 in the very general *kernel probabilistic programming* framework and obtain a consistent estimator of the KME $\mu_{f(X)}$ of $f(X)$ for *any* continuous function f, there is a technical condition that the L_1 norm $\sum_{m=1}^{M} |w_m|$ of the released weights may need to remain bounded by a constant as $N \to \infty$ (Simon-Gabriel et al., 2016). This is not enforced by Algorithm 1, but Theorem 11 in Appendix A.1 shows how a simple regularisation in the final stage of the algorithm achieves this without breaking consistency (or privacy).

The next result about Algorithm 1 shows that it is differentially private whenever $k(x, x) \leq 1$ for all $x \in \mathcal{X}$. This is a weak assumption that holds for all normalised kernels, and can be achieved by simple rescaling for any bounded kernel (such that $\sup_{x \in \mathcal{X}} k(x, x) < \infty$). When $\mathcal{X}$ is a compact domain, all continuous kernels are bounded.

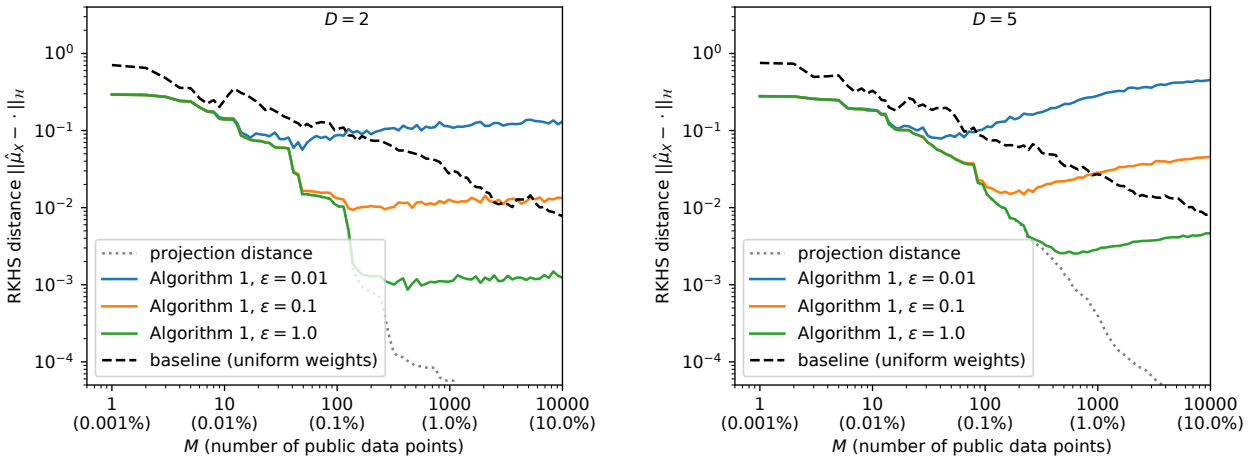

Figure 1: RKHS distance (lower is better) to the (private) empirical KME $\hat{\mu}_X$ computed using the entire private database of size $N = 100,000$. The dimension of the database was $D = 2$ (left) or $D = 5$ (right); please see Appendix B for further details of the setup. Horizontally we varied M, the number of publicly releasable data points. Stricter privacy requirements (lower ε) naturally lead to lower accuracy. Increasing M does not always necessarily improve accuracy, since a new public data point always increases the total amount of privatising noise that needs to be added, but this might not be outweighed by its positive contribution towards covering relevant parts of the input space. In all cases, for sufficiently small M Algorithm 1 provided a significantly more accurate estimate than μ^{baseline}.

Proposition 3. *If $k(x, x) \leq 1$ for all $x \in \mathcal{X}$, then Algorithm 1 is (ε, δ)-differentially private.*

Remark 4. One usually requires that δ decreases faster than polynomially with the database size N (Dwork & Roth, 2014). The proof of Theorem 2 remains valid whenever $M(N) \in o(N^2 / \ln(1.25/\delta(N)))$, so for example we could have $\delta(N) = e^{-\sqrt{N}}$ and $M(N) \in o(N^{3/2})$. □

For a finite private database, actual performance will heavily depend on how the synthetic data points are chosen. We consider the following two extreme scenarios:

1. *No publishable subset*: No rows of the private database are, or can be made public unmodified.

2. *Publishable subset*: A small proportion of the private database is already public, or can be made public.

Proposition 5 (Algorithm 1, No publishable subset)**.** *Say $\mathcal{X}$ is a bounded subset of $\mathbb{R}^D$, the kernel k is Lipschitz, and the synthetic data points $z_1, z_2, \ldots$ are sampled i.i.d. from a distribution q with density bounded away from 0 on any bounded subset of $\mathbb{R}^D$. Then $M = M(N)$ can be chosen so that the output of Algorithm 1 converges to the true KME μ_X in RKHS norm at a rate $\mathcal{O}_p(N^{-1/(D+1+c)})$, where c is any fixed positive number $c > 0$.*

Unsurprisingly, the convergence rate deteriorates with input dimension D, since without prior information about the private data manifold it is increasingly difficult for randomly sampled synthetic points to capture patterns in the private data. One of the main strengths of KMEs is that the empirical estimator converges to the true embedding at a rate $\mathcal{O}_p(N^{-1/2})$ independently of the input dimension D, so we see that in this unfavourable scenario Algorithm 1 incurs a substantial privacy cost in high dimensions. On the other hand, if a small, but fixed proportion of the private database is publishable, then Algorithm 1 incurs no privacy cost in terms of the convergence rate:

Proposition 6 (Algorithm 1, Publishable subset)**.** *Say that a fixed proportion η of the private database can be published unmodified. Using this part of the database as the synthetic data points, Algorithm 1 outputs a consistent estimator of μ_X that converges in RKHS norm at a rate $\mathcal{O}_p(N^{-1/2})$.*

Note that in this scenario the rate $\mathcal{O}_p(N^{-1/2})$ can be also achieved by uniform weighting of the synthetic data points, since $\hat{\mu}^{\text{baseline}} := \frac{1}{M}\sum_{m=1}^{M} k(z_m, \cdot)$ with $z_m = x_m$ is already a consistent estimator of μ_X (although based on a much smaller sample size $M = \eta N \ll N$). The purpose of Algorithm 1 is to find (generally non-uniform) $w_1, \ldots, w_M$ that reweight the public data points using the information in the large private dataset, but respecting differential privacy. Proposition 6 confirmed theoretically that this does not hurt the convergence rate, while Figure 1 shows empirically on two synthetic datasets of dimensions $D = 2$ and $D = 5$ that Algorithm 1 can in fact yield more accurate estimates of the KME than $\hat{\mu}^{\text{baseline}}$, especially when the proportion of public data points is small. This is encouraging, since obtaining permission to publish a larger subset of the private data unchanged will usually come at an increased cost. The ability to instead reweight a smaller public dataset in a differentially private manner using Algorithm 1 is therefore useful.

Algorithm 2 Differentially private database release via a random features RKHS

Input: database $D = \{x_1, \ldots, x_N\} \subseteq \mathcal{X}$, kernel k on $\mathcal{X}$, privacy parameters $\varepsilon > 0$ and $\delta > 0$
Output: (ε, δ)-differentially private, weighted synthetic database (representing an estimate of μ_X in the RKHS $\mathcal{H}$ of k)
1: $J \leftarrow J(N) \in \omega(1) \cap o(N^2)$, number of random features to use
2: $\phi \leftarrow$ random feature map $\mathcal{X} \mapsto \mathbb{R}^J$ for kernel k with J features
3: $\hat{\mu}_X^\phi \leftarrow \frac{1}{N}\sum_{n=1}^N \phi(x_n) \in \mathbb{R}^J$, empirical KME of X in RKHS $\mathcal{H}_\phi$ of the random features kernel $k_\phi(\cdot,\cdot) := \phi(\cdot)^T\phi(\cdot)$
4: $\tilde{\mu}_X^\phi \leftarrow \hat{\mu}_X^\phi + \mathcal{N}(0, \frac{8\ln(1.25/\delta)}{N^2\varepsilon^2} I_J)$, an (ε, δ)-differentially private version of the vector $\hat{\mu}_X^\phi$ (Gaussian mechanism)
5: $M \leftarrow M(N) \geq N$, number of synthetic expansion points to use for representing $\tilde{\mu}_X^\phi$
6: $(z_1, w_1), \ldots, (z_M, w_M) \leftarrow$ approximate $\tilde{\mu}_X^\phi$ in the RKHS $\mathcal{H}_\phi$ using a Reduced set method:

$$(z_1, w_1), \ldots, (z_M, w_M) \approx \underset{(z'_1,w'_1),\ldots,(z'_M,w'_M) \text{ s.t. } \sum_m |w'_m| \leq 1}{\operatorname{argmin}} \left\| \sum_{m=1}^M w'_m \phi(z'_m) - \tilde{\mu}_X^\phi \right\|_{\mathcal{H}_\phi} \tag{5}$$

7: **return** $(z_1, w_1), \ldots, (z_M, w_M)$

5. Perturbation in Random-Features RKHS

Another approach to ensuring differential privacy is to map the potentially infinite dimensional RKHS $\mathcal{H}$ of k into a different, finite-dimensional RKHS $\mathcal{H}_\phi$ using random features (Rahimi & Recht, 2007), privacy-protect the finite coordinate vector in this space (Chaudhuri et al., 2011), and then employ a reduced set method to find an expansion of the resulting RKHS element in terms of synthetic data points. In contrast to Algorithm 1, both the weights and locations of synthetic data points can be optimised here.

The algorithm is formalised as Algorithm 2 above. Lines 1-2 pick the number $J = J(N)$ of random features to use, and construct a random feature map ϕ with that many features. Lines 3-4 compute the empirical KME of X in the RKHS $\mathcal{H}_\phi$ corresponding to the kernel induced by the random features, and then privacy-protect the resulting finite, real-valued vector. Lines 5-6 run a blindly initialised Reduced set method to find a weighted synthetic dataset whose KME in $\mathcal{H}_\phi$ is close to the privacy-protected KME of the private database. Line 7 releases this weighted dataset to the public.

The following theorem confirms that Algorithm 2 outputs a consistent estimator of the true KME μ_X, provided that the optimisation problem (5) is solved exactly, and the random features converge to the kernel k uniformly on $\mathcal{X}$. On compact sets $\mathcal{X}$ this requirement is satisfied by general schemes such as *random Fourier features* and *random binning* for shift-invariant kernels (Rahimi & Recht, 2007), or by random features for dot product kernels (Kar & Karnick, 2012).

Theorem 7. *If $\phi(\cdot)^T\phi(\cdot) \to k(\cdot,\cdot)$ converges uniformly in $\mathcal{X} \times \mathcal{X}$ as $J \to \infty$, then the output of Algorithm 2 is a consistent estimator of the true KME μ_X as $N \to \infty$.*

Moreover, a uniform convergence rate for the random features, such as the one for random Fourier features by Sriperumbudur & Szabo (2015), can be used to derive a convergence rate for the output of Algorithm 2:

Proposition 8. *If $\phi(\cdot)^T\phi(\cdot) \to k(\cdot,\cdot)$ converges uniformly in $\mathcal{X} \times \mathcal{X}$ at a rate $\mathcal{O}_p(J^{-1/2})$ as $J \to \infty$, then $J = J(N)$ can be chosen so that the output of Algorithm 2 converges to the true KME μ_X at a rate $\mathcal{O}_p(N^{-1/3})$.*

The empirical KME of the private database $\hat{\mu}_X$ converges at a rate $\mathcal{O}_p(N^{-1/2})$, so we see that under perfect optimisation, the privacy cost incurred by Algorithm 2 is a factor of $N^{1/6}$. In practice performance will also depend on the Reduced set method used, and the computational budget allocated to it. Figure 2 shows how the incurred error (in terms of RKHS distance) varies with the number of synthetic data points M. The additional ability of Algorithm 2 to optimise the *locations* of the synthetic data points (rather than just the weights, as in Algorithm 1) seems to be more helpful in the higher-dimensional case $D = 5$, where the randomly sampled synthetic data points are less likely to land close to private data points.

Proposition 9. *If $\|\phi(x)\|_2 \leq 1$ for all $x \in \mathcal{X}$, then Algorithm 2 is (ε, δ)-differentially private.*

This L_2-boundedness requirement on the random feature vectors $\phi(x)$ is reasonable under the weak assumption $k(x,x) \leq 1$ for all $x \in \mathcal{X}$ discussed in Section 4, as in that case $\|\phi(x)\|_2^2 = \phi(x)^T\phi(x) \approx k(x,x) \leq 1$.

6. Related Work

Synthetic database release algorithms with a differential privacy guarantee have been studied in the literature before. Machanavajjhala et al. (2008) analyzed such a procedure for count data, ensuring privacy by sampling a distribution and then synthetic counts from a Dirichlet-Multinomial posterior. Blum et al. (2008) studied the exponential mechanism applied to synthetic database generation, which leads to a very general, but unfortunately inefficient algorithm (see also Section 3.4). Wasserman & Zhou (2010) provided a theoretical comparison of this algorithm to sampling synthetic

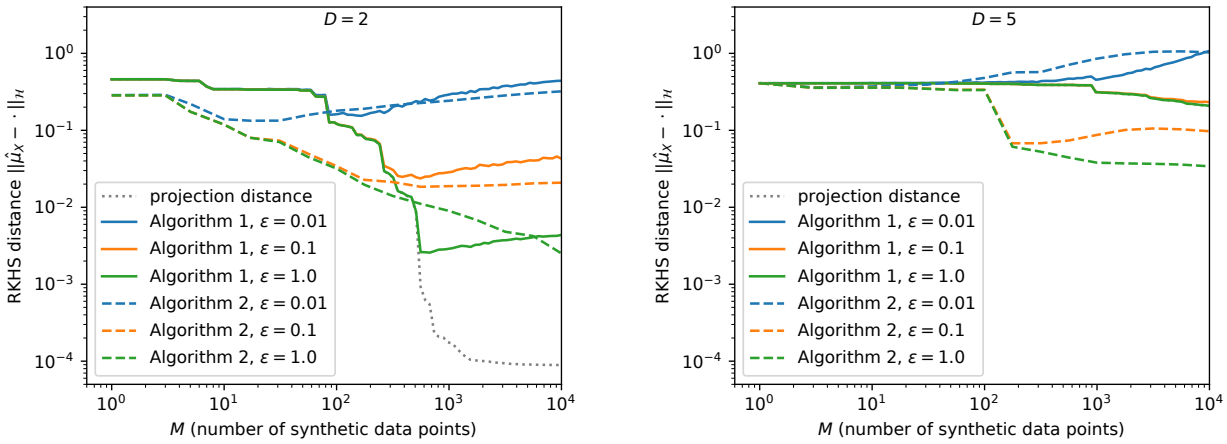

Figure 2: RKHS distance (lower is better) to the (private) empirical KME $\hat{\mu}_X$ computed using the same databases as in Figure 1, of dimensions $D = 2$ (left) and $D = 5$ (right), but this time without a publishable subset. The synthetic data points for Algorithm 1 were therefore sampled from a wide Gaussian distribution; please see Appendix B for further details. Algorithm 2 is capable of outperforming Algorithm 1 thanks to its ability to optimise the synthetic data point locations, but this depends on the precise optimisation procedure used and the optimisation problem becomes harder in higher dimensions.

databases from deterministically smoothed, or randomly perturbed histograms. Unlike our approach, these algorithms achieve differential privacy by sampling synthetic data points from a specific distribution, where resorting to approximate sampling can break the privacy guarantee. In our framework we propose to arrive at the synthetic database using a reduced set method, where poor performance could affect statistical usefulness of the synthetic database, but cannot break its differential privacy.

Zhou et al. (2009) and Kenthapadi et al. (2012) proposed randomised database compression schemes that yield synthetic databases useful for particular types of algorithms, while guaranteeing differential privacy. The former compresses the number of data points using a random linear or affine transformation of the entire database, and the result can be used by procedures that rely on the empirical covariance of the original data. The latter compresses the number of data point dimensions while approximately preserving distances between original, private data points.

Differentially private learning in a RKHS has also been studied, with Chaudhuri et al. (2011) and Rubinstein et al. (2012) having independently presented release mechanisms for the result of an empirical risk minimisation procedure (such as a SVM). Similarly to our Algorithm 2, they map data points into a finite-dimensional space defined by random features and carry out the privacy-protecting perturbation there. However, they do not require the final stage of invoking a Reduced set method to construct a synthetic database, because the output (such as a trained SVM) is only used for evaluation on test points, for which it suffices to additionally release the used random feature map ϕ.

As our framework stipulates privacy-protecting an empirical KME, which is a function $\mathcal{X} \to \mathbb{R}$, the work on differential privacy for functional data is of relevance. Hall et al. (2013) showed how an RKHS element can be made differentially private via pointwise addition of a Gaussian process sample path, but as discussed in Section 3.2, the resulting function is no longer an element of the RKHS. Recently, Aldà & Rubinstein (2017) proposed a general Bernstein mechanism for ε-differentially private function release. The released function can be evaluated pointwise arbitrarily many times, but again, the geometry of the RKHS to which the unperturbed function belonged cannot be easily exploited anymore.

7. Discussion

We proposed a framework for constructing differentially private synthetic database release algorithms, based on the idea of using KMEs in RKHS as intermediate database representations. To justify our framework, we presented two concrete algorithms and proved theoretical results guaranteeing their consistency and differential privacy. We also studied their finite-sample convergence rates, and provided empirical illustrations of their performance on synthetic datasets. We believe that exploring other instantiations of this framework, and comparing them theoretically and empirically, can be a fruitful direction for future research.

The i.i.d. assumption on database rows can be relaxed. For example, if they are identically distributed (as a random variable X), but not necessarily independent, the framework remains valid as long as a consistent estimator of the KME μ_X can be constructed from the database rows. A common situation where this arises is, for example, duplication of database rows due to user error.

Acknowledgements

The authors would like to thank Bharath Sriperumbudur and the anonymous reviewers for helpful feedback.

References

Aldà, F. and Rubinstein, B. I. P. The Bernstein mechanism: Function release under differential privacy. In *AAAI*, 2017.

Blum, A., Ligett, K., and Roth, A. A learning theory approach to non-interactive database privacy. In *40th ACM Symposium on Theory of Computing*, 2008.

Burges, C. J. C. Simplified support vector decision rules. In *ICML*, 1996.

Chaudhuri, K., Monteleoni, C., and Sarwate, A. D. Differentially private empirical risk minimization. *JMLR*, 12, 2011.

Chen, Y., Welling, M., and Smola, A. Super-samples from kernel herding. In *UAI*, 2010.

Dwork, C. Differential privacy. In *33rd International Conference on Automata, Languages and Programming (ICALP)*, 2006.

Dwork, C. and Roth, A. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9, 2014.

Fukumizu, K., Gretton, A., Sun, X., and Schölkopf, B. Kernel measures of conditional dependence. In *NIPS*, 2008.

Gretton, A., Bousquet, O., Smola, A., and Schölkopf, B. Measuring statistical dependence with Hilbert-Schmidt norms. In *ALT*, 2005.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *JMLR*, 13, 2012.

Hall, R., Rinaldo, A., and Wasserman, L. Differential Privacy for Functions and Functional Data. *JMLR*, 14, 2013.

Kar, P. and Karnick, H. Random feature maps for dot product kernels. In *AISTATS*, 2012.

Kenthapadi, K., Korolova, A., Mironov, I., and Mishra, N. Privacy via the Johnson-Lindenstrauss transform. *arXiv:1204.2606 [cs]*, 2012.

Lopez-Paz, D., Muandet, K., Schölkopf, B., and Tolstikhin, I. Towards a learning theory of cause-effect inference. In *ICML*, 2015.

Machanavajjhala, A., Kifer, D., Abowd, J., Gehrke, J., and Vilhuber, L. Privacy: Theory meets practice on the map. In *IEEE 24th International Conference on Data Engineering*, 2008.

McSherry, F. and Talwar, K. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2007.

Muandet, K., Sriperumbudur, B., Fukumizu, K., Gretton, A., and Schölkopf, B. Kernel mean shrinkage estimators. *JMLR*, 17, 2016.

Rahimi, A. and Recht, B. Random features for large scale kernel machines. In *NIPS*, 2007.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

Rubinstein, B. I. P., Bartlett, P. L., Huang, L., and Taft, N. Learning in a large function space: Privacy-preserving mechanisms for SVM learning. *The Journal of Privacy and Confidentiality*, 4 (1), 2012.

Schölkopf, B. and Smola, A. J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

Schölkopf, B., Muandet, K., Fukumizu, K., Harmeling, S., and Peters, J. Computing functions of random variables via Reproducing Kernel Hilbert Space representations. *Statistics and Computing*, 25, 2015.

Simon-Gabriel, C.-J., Ścibior, A., Tolstikhin, I., and Schölkopf, B. Consistent Kernel Mean Estimation for Functions of Random Variables. In *NIPS*, 2016.

Smola, A., Gretton, A., Song, L., and Schölkopf, B. A Hilbert space embedding for distributions. In *ALT*, 2007.

Sriperumbudur, B. and Szabo, Z. Optimal rates for random Fourier features. In *NIPS*. 2015.

Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. G. Universality, characteristic kernels and RKHS embedding of measures. *JMLR*, 2011.

Tolstikhin, I., Sriperumbudur, B. K., and Muandet, K. Minimax estimation of kernel mean embeddings. *JMLR*, 18, 2017.

Wasserman, L. and Zhou, S. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105, 2010.

Zhou, S., Ligett, K., and Wasserman, L. Differential privacy with compression. In *IEEE International Conference on Symposium on Information Theory (ISIT)*, 2009.