J Cryptol (2020) 33:2078–2112
https://doi.org/10.1007/s00145-020-09363-y

Journal of
CRYPTOLOGY

PCPs and the Hardness of Generating Synthetic Data*

Jonathan Ullman
Khoury College of Computer Science, Northeastern University, Boston, MA, USA
jullma@ccs.neu.edu

Salil Vadhan
Harvard John A. Paulson School of Engineering and Applied Sciences, Cambridge, MA, USA
salil_vadhan@harvard.edu

Communicated by Ran Canetti.

Received 8 August 2014 / Revised 1 July 2020
Online publication 31 July 2020

Abstract. Assuming the existence of one-way functions, we show that there is no polynomial-time differentially private algorithm $\mathcal{A}$ that takes a database $D \in (\{0,1\}^d)^n$ and outputs a "synthetic database" $\hat{D}$ all of whose two-way marginals are approximately equal to those of D. (A two-way marginal is the fraction of database rows $x \in \{0,1\}^d$ with a given pair of values in a given pair of columns.) This answers a question of Barak et al. (PODS '07), who gave an algorithm running in time $\mathrm{poly}(n, 2^d)$. Our proof combines a construction of hard-to-sanitize databases based on digital signatures (by Dwork et al., STOC '09) with encodings based on the PCP theorem. We also present both negative and positive results for generating "relaxed" synthetic data, where the fraction of rows in D satisfying a predicate c are estimated by applying c to each row of $\hat{D}$ and aggregating the results in some way.

1. Introduction

There are many settings in which it is desirable to share information about a database that contains sensitive information about individuals. For example, doctors may want to share information about health records with medical researchers, the federal government may want to release census data for public information, and a company like Netflix may want to provide its movie rental database for a public competition to develop a better recommendation system. However, it is important to do this in way that preserves the "privacy" of the individuals whose records are in the database. *Differential privacy* has emerged over the past decade as a robust and mathematically rigorous definition of individual privacy (see the textbook [17]).

*A preliminary version of this work appeared in the Theory of Cryptography Conference 2011.

J. Ullman: This work was done while the author was in the Harvard John A. Paulson School of Engineering and Applied Sciences. Supported by NSF Grant CNS-0831289.
S. Vadhan: Supported by NSF Grant CNS-0831289.

Differential Privacy. A randomized algorithm $\mathcal{A}$ is defined to be *differentially private* [13] if for every two databases $D = (x_1, \ldots, x_n)$, $D' = (x'_1, \ldots, x'_n)$ that differ on exactly one row, the distributions $\mathcal{A}(D)$ and $\mathcal{A}(D')$ are "close" to each other. Formally, we require that $\mathcal{A}(D)$ and $\mathcal{A}(D')$ assign the same probability mass to every event, up to a multiplicative factor of $e^{\varepsilon} \approx 1 + \varepsilon$, where ε is typically taken to be a small constant. (In addition to this multiplicative factor, the probabilities are often allowed to differ by a negligible additive term.) This captures the idea that no individual's data have a significant influence on the output of $\mathcal{A}$ (provided that data about an individual is confined to one or a few rows of the database). Differential privacy has several nice properties, such as being agnostic to the adversary's prior information and degrading smoothly under composition.

With this model of privacy, the goal becomes to design algorithms $\mathcal{A}$ that simultaneously meet the above privacy guarantee and give "useful" information about the database. For example, we may have a true query function c in which we are interested, and the goal is to design $\mathcal{A}$ that is differentially private (with ε as small as possible) and estimates c well (e.g., the error $|\mathcal{A}(D) - c(D)|$ is small with high probability). For example, if $c(D)$ is the fraction of database rows that satisfy some property—a *counting query*—then it is known that we can take $\mathcal{A}(D)$ to equal $c(D)$ plus random Laplacian noise with standard deviation $O(1/(\varepsilon n))$, where n is the number of rows in the database and ε is the measure of differential privacy [6]. A sequence of works [6,10,13,16] has provided a very good understanding of differential privacy in an interactive model in which real-valued queries c are made and answered one at a time. The amount of noise that one needs when responding to a query c should be based on the sensitivity of c, as well as the total number of queries answered so far.

However, for many applications, it would be more attractive to do a noninteractive data release, where we compute and release a single, differentially private "summary" of the database that enables others to determine accurate answers to a large class of queries. What form should this summary take? The most appealing form would be a *synthetic database*, which is a new database $\widehat{D} = \mathcal{A}(D)$ whose rows are "fake," but come from the same universe as those of D and are guaranteed to share many statistics with those of D (up to some accuracy). Some advantages of synthetic data are that it can be easily understood by humans, and statistical software can be run directly on it without modification. For example, these considerations led the German Institute for Employment Research to adopt synthetic databases for the release of employment statistics [33].

Previous Results on Synthetic Data. The first result on producing differentially private synthetic data came in the work of Barak et al. [3]. Given a database D consisting of n rows from $\{0, 1\}^d$, they show how to construct a differentially private synthetic database $\widehat{D}$, also of n rows from $\{0, 1\}^d$, in which the full "contingency table," consisting of all conjunctive counting queries, is approximately preserved. That is, for every conjunction $c(x_1, \ldots, x_n) = x_{i_1} \wedge x_{i_2} \wedge \cdots \wedge x_{i_k}$ for $i_1, \ldots, i_k \in [d]$, the fraction of rows in $\widehat{D}$ that satisfy c equals the fraction of rows in D that satisfy c up to an additive error of $2^{O(d)}/n$. The running time of their algorithm is $\mathrm{poly}(n, 2^d)$, which is feasible for small values of d. They pose as an open problem whether the running time of their algorithm can be improved for the case where we only want to preserve the *k-way marginals* for small k (e.g., $k = 2$). These are the counting queries corresponding to conjunctions of up to k literals. Indeed, there are only $O(d)^k$ such conjunctions, and we can produce differentially private estimates for all the corresponding counting queries in time $\text{poly}(n, d^k)$ by just adding noise $O(d)^k/n$ to each one. Moreover, a version of the Barak et al. algorithm [3] can ensure that even these noisy answers are consistent with a real database.[1]

A more general and dramatic illustration of the potential expressiveness of synthetic data came in the work of Blum et al. [7]. They show that for every class $\mathcal{C} = \{c : \{0,1\}^d \rightarrow \{0,1\}\}$ of predicates, there is a differentially private algorithm A that produces a synthetic database $\hat{D} = \mathcal{A}(D)$ such that all counting queries corresponding to predicates in $\mathcal{C}$ are preserved to within an accuracy of $\tilde{O}((d \log(|\mathcal{C}|)/n)^{1/3})$, with high probability. In particular, with $n = \text{poly}(d)$, the synthetic data can provide simultaneous accuracy for an *exponential-sized* family of queries (e.g., $|\mathcal{C}| = 2^d$). Unfortunately, the running time of the BLR mechanism is also exponential in d.

Dwork et al. [14] gave evidence that the large running time of the BLR mechanism is inherent. Specifically, assuming the existence of one-way functions, they exhibit an efficiently computable family $\mathcal{C}$ of predicates (e.g., consisting of circuits of size d^2) for which it is infeasible to produce a differentially private synthetic database preserving the counting queries corresponding to $\mathcal{C}$ (for databases of any $n = \text{poly}(d)$ number of rows). For non-synthetic data, they show a close connection between the infeasibility of producing a differentially private summarization and the existence of efficient "traitor-tracing schemes." However, these results leave open the possibility that for *natural* families of counting queries (e.g., those corresponding to conjunctions), producing a differentially private synthetic database (or non-synthetic summarization) can be done efficiently. Indeed, one may have gained optimism from a comparison with the early history of computational learning theory, where one-way functions were used to show hardness of learning arbitrary efficiently computable concepts in computational learning theory, but natural subclasses (like conjunctions) were found to be learnable [38].

Our Results. We prove that it is infeasible to produce synthetic databases preserving even very simple counting queries, such as 2-way marginals:

Theorem 1.1. *Assuming the existence of one-way functions, there is a constant $\gamma > 0$ such that for every polynomial p, there is no polynomial-time, differentially private algorithm $\mathcal{A}$ that takes a database $D \in (\{0,1\}^d)^{p(d)}$ and produces a synthetic database $\hat{D} \in (\{0,1\}^d)^*$ such that $|c(D) - c(\hat{D})| \leq \gamma$ for all 2-way marginals c.*[2]

Our result actually shows that any computationally efficient algorithm that generates synthetic data approximately preserving all 2-way marginals of this dataset must, in the worst case, include one of the rows of the original dataset in the synthetic dataset. In particular, this result holds when the data consist of n independent samples from a specific distribution with high min-entropy, so *a priori* an attacker would have a negligible probability of recovering the row. Therefore, our result rules out any notion

[1] Technically, this "real database" may assign fractional weight to some rows.

[2] Recall that a 2-way marginal $c(D)$ computes the fraction of database rows satisfying a conjunction of two literals, i.e., the fraction of rows $x_i \in \{0,1\}^d$ such that $x_{i,j} = b$ and $x_{i,j'} = b'$ for some columns $j, j' \in [d]$ and values $b, b' \in \{0,1\}$.

of privacy that prevents an attacker from efficiently recovering an entire row of a dataset with high min-entropy.

Our impossibility result also extends from conjunctions of 2 literals to any family of constant-arity predicates containing a function of at least two variables, such as parities of 3 literals.

As mentioned earlier, all 2-way marginals *can* be easily summarized with non-synthetic data (by just adding noise to each of the $(2d)^2$ values). Thus, our result shows that requiring a synthetic database may severely constrain what sorts of differentially private data releases are possible. (Dwork et al. [14] also showed that there exists a $\text{poly}(d)$-sized family of counting queries that are hard to summarize with synthetic data, thereby separating synthetic data from non-synthetic data. Our contribution is to show that such a separation holds for a very simple and natural family of predicates, namely 2-way marginals.)

This separation between synthetic data and non-synthetic data seems analogous to the separations between proper and improper learning in computational learning theory [19,32], where it is infeasible to learn certain concept classes if the output hypothesis is constrained to come from the same representation class as the concept, but it becomes feasible if we allow the output hypothesis to come from a different representation class. This analogy gives hope for designing efficient, differentially private algorithms that take a database and produce a compact summary of it that is not synthetic data but somehow can be used to accurately answer exponentially many questions about the original database (e.g., all marginals). The negative results of [14] on non-synthetic data (assuming the existence of efficient traitor-tracing schemes) do not say anything about natural classes of counting queries, such as marginals.

To bypass the complexity barrier stated in Theorem 1.1, it may not be necessary to introduce exotic data representations; some mild generalizations of synthetic data may suffice. For example, several recent algorithms [7,18,35] produce several synthetic databases, with the guarantee that the *median* answer over these databases is approximately accurate. More generally, we can consider summarizations of a database D that that consist of a collection $\hat{D}$ of rows from the same universe as the original database, and where we estimate $c(D)$ by applying the predicate c to each row of $\hat{D}$ and then aggregating the results via some aggregation function f. With standard synthetic data, f is simply the average, but we may instead allow f to take a median of averages, or apply an affine shift to the average. For such *relaxed synthetic data*, we prove the following results:

- There is a constant k such that counting queries corresponding to k-juntas (functions depending on at most k variables) *cannot* be accurately and privately summarized as relaxed synthetic data with a median-of-averages aggregator, or with a symmetric and monotone aggregator (that is independent of the predicate c being queried).
- For every constant k, counting queries corresponding to k-juntas *can* be accurately and privately summarized as relaxed synthetic data with an aggregator that applies an affine shift to the average (where the shift does depend on the predicate being queried).

We remark that under stronger (but still plausible) cryptographic assumptions, we can show that generating private synthetic data requires almost exponential time (time

$2^{d^{1-o(1)}}$). Subsequent to our work, algorithms running in subexponential time ($2^{d^{1-\varepsilon}}$ for some constant $\varepsilon > 0$) have been given using non-synthetic data approaches inspired by non-proper learning algorithms for a regime where adding independent noise does not work (the number of queries in the family is much larger than n^2) [8,15,22,37].

Techniques. Our proof of Theorem 1.1 and our other negative results are obtained by combining the hard-to-sanitize databases of Dwork et al. [14] with probabilistically checkable proofs (PCPs). They construct a database consisting of valid message–signature pairs (m_i, σ_i) under a digital signature scheme and argue that any differentially private sanitizer that preserves accuracy for the counting query associated with the signature verification predicate can be used to forge valid signatures. We replace each message–signature pair (m_i, σ_i) with a PCP encoding π_i that proves that (m_i, σ_i) satisfies the signature verification algorithm. We then argue that if accuracy is preserved for a large fraction of the (constant arity) constraints of the PCP verifier, then we can "decode" the PCP either to violate privacy (by recovering one of the original message–signature pairs) or to forge a signature (by producing a new message–signature pair).

We remark that error-correcting codes were already used in [14] for the purpose of producing a fixed polynomial-sized set of counting queries that can be used for all verification keys. Our observation is that by using *PCP encodings*, we can reduce not only the number of counting queries in consideration, but also their computational complexity.

Our proof has some unusual features among PCP-based hardness results:

- As far as we know, this is the first time that PCPs have been used in conjunction with cryptographic assumptions for a hardness result. (They have been used together for positive results regarding computationally sound proof systems [4,27,29].) It would be interesting to see if such a combination could be useful in, say, computational learning theory (where PCPs have been used for hardness of "proper" learning [1,20] and cryptographic assumptions for hardness of representation-independent learning [25,38]).
- While PCP-based inapproximability results are usually stated as Karp reductions, we actually need them to be *Levin* reductions—capturing that they are reductions between search problems, and not just decision problems. (Previously, this property has been used in the same results on computationally soundproofs mentioned above.)

2. Preliminaries

2.1. *Sanitizers*

Let a *database* $D \in (\{0,1\}^d)^n$ be a matrix of n rows, $x_1, \ldots, x_n$, corresponding to people, each of which contains d binary attributes. A *sanitizer* $\mathcal{A} : (\{0,1\}^d)^n \to \mathcal{R}$ takes a database and outputs some data structure in $\mathcal{R}$. In the case where $\mathcal{R} = (\{0,1\}^d)^{\hat{n}}$ (an $\hat{n}$-row database), we say that $\mathcal{A}$ outputs a *synthetic database*.

We would like such sanitizers to be both *private* and *accurate*. In particular, the notion of privacy we are interested in is as follows

Definition 2.1. *(Differential Privacy)* [13] A sanitizer $\mathcal{A} : (\{0,1\}^d)^n \to \mathcal{R}$ is (ε, δ)-*differentially private* if for every two databases $D_1, D_2 \in (\{0,1\}^d)^n$ that differ on exactly one row, and every subset $S \subseteq \mathcal{R}$

$$\Pr[\mathcal{A}(D_1) \in S] \le e^{\varepsilon} \Pr[\mathcal{A}(D_2) \in S] + \delta$$

In the case where $\delta = 0$, we say that $\mathcal{A}$ is ε-*differentially private*.

Since a sanitizer that always outputs 0 satisfies Definition 2.1, we also need to define what it means for a database to be accurate. In this paper, we consider accuracy with respect to counting queries. Consider a set $\mathcal{C}$ consisting of Boolean predicates $c : \{0,1\}^d \to \{0,1\}$, which we call a *concept class*. Then each predicate c induces a *counting query* that on database $D = (x_1, \ldots, x_n) \in (\{0,1\}^d)^n$ returns

$$c(D) = \frac{1}{n} \sum_{i=1}^{n} c(x_i)$$

If the output of $\mathcal{A}$ is a synthetic database $\widehat{D} \in (\{0,1\}^d)^*$, then $c(\mathcal{A}(D))$ is simply the fraction of rows of $\widehat{D}$ that satisfy the predicate c. However, if $\mathcal{A}$ outputs a data structure that is not a synthetic database, then we require that there is an *evaluator* function $\mathcal{E} : \mathcal{R} \times \mathcal{C} \to \mathbb{R}$ that estimates $c(D)$ from the output of $\mathcal{A}(D)$ and the description of c. For example, $\mathcal{A}$ may output a vector $Z = (c(D) + Z_c)_{c \in \mathcal{C}}$ where Z_c is a random variable for each $c \in \mathcal{C}$, and $\mathcal{E}(Z, c)$ is the cth component of $Z \in \mathcal{R} = \mathbb{R}^{|\mathcal{C}|}$. Abusing notation, we will write $c(Z)$ and $c(\mathcal{A}(D))$ as shorthand for $\mathcal{E}(Z, c)$ and $\mathcal{E}(\mathcal{A}(D), c)$, respectively.

We will say that $\mathcal{A}$ is "accurate" for the concept class $\mathcal{C}$ if the fractional counts $c(\mathcal{A}(D))$ are close to the fractional counts $c(D)$. Formally

Definition 2.2. *(Accuracy)* An output Z of sanitizer $\mathcal{A}(D)$ is α-*accurate* for a concept class $\mathcal{C}$ if

$$\forall c \in \mathcal{C}, |c(Z) - c(D)| \le \alpha.$$

A sanitizer $\mathcal{A}$ is (α, β)-*accurate* for a concept class $\mathcal{C}$ if for every database D,

$$\Pr_{\mathcal{A}'s \text{ coins}} [\forall c \in \mathcal{C},\ |c(\mathcal{A}(D)) - c(D)| \le \alpha] \ge 1 - \beta$$

In this paper, we write $f(n) = \mathrm{negl}(n)$ if $f(n) = o(n^{-c})$ for every $c > 0$ and say that $f(n)$ is *negligible*. We use $|s|$ to denote the length of the string s, and $s_1 \| s_2$ to denote the concatenation of s_1 and s_2.

2.2. *Hardness of Sanitizing*

Differential privacy is a very strong notion of privacy, so it is common to look for hardness results that rule out weaker notions of privacy. These hardness results show that every sanitizer must be "blatantly non-private" in some sense. In this paper, our notion of blatant non-privacy roughly states that there exists an efficient adversary who can find a row of the original database using only the output from any efficient sanitizer. Such definitions are also referred to as “row non-privacy.” We define hardness-of-sanitization with respect to a particular concept class and want to exhibit a distribution on databases for which it would be infeasible for any efficient sanitizer to give accurate output without revealing a row of the database. Specifically, following [14], we define the following notions

Definition 2.3. *(Database Distribution Ensemble)* Let $\mathcal{D} = \mathcal{D}_d$ be an ensemble of distributions on d-column databases with $n+1$ rows $D \in (\{0,1\}^d)^{n+1}$. Let $(D, D', i) \leftarrow_{\mathrm{R}} \tilde{\mathcal{D}}$ denote the experiment in which we choose $D_0 \leftarrow_{\mathrm{R}} \mathcal{D}$ and $i \in [n]$ uniformly at random, and set D to be the first n rows of D_0 and D' to be D with the ith row replaced by the $(n+1)$-st row of D_0.

Definition 2.4. *(Hard-to-sanitize Distribution)* Let $\mathcal{C}$ be a concept class, $\alpha : \mathbb{N} \to [0,1]$ and $T_{\mathrm{San}} : \mathbb{N} \to \mathbb{N}$ be functions, and $\mathcal{D} = \mathcal{D}_d$ be a database distribution ensemble.

The distribution $\mathcal{D}$ is $(\alpha, T_{\mathrm{San}}, \mathcal{C})$*-hard-to-sanitize* if there exists an efficient adversary $\mathcal{T}$ such that for any alleged sanitizer $\mathcal{A}$ running in time at most $T_{\mathrm{San}}(d)$ the following two conditions hold:

1. Whenever $\mathcal{A}(D)$ is $\alpha(d)$-accurate, then $\mathcal{T}(\mathcal{A}(D))$ outputs a row of D:

$$\Pr_{\substack{(D,D',i)\leftarrow_{\mathrm{R}}\tilde{\mathcal{D}}\\ \mathcal{A}'s \text{ and } \mathcal{T}'s \text{ coins}}} [(\mathcal{A}(D) \text{ is } \alpha(d)\text{-accurate for } \mathcal{C}) \wedge (\mathcal{T}(\mathcal{A}(D)) \cap D = \emptyset)] \le \mathrm{negl}(d).$$

2. For every efficient sanitizer $\mathcal{A}$, $\mathcal{T}$ cannot extract x_i from the database D':

$$\Pr_{\substack{(D,D',i)\leftarrow_{\mathrm{R}}\tilde{\mathcal{D}}\\ \mathcal{A}'s \text{ and } \mathcal{T}'s \text{ coins}}} \left[\mathcal{T}(\mathcal{A}(D')) = x_i\right] \le \mathrm{negl}(d)$$

where x_i is the ith row of D.

In [14], it was shown that every distribution that is $(\alpha, T_{\mathrm{San}}, \mathcal{C})$-hard-to-sanitize in the sense of Definition 2.4 is also hard to sanitize while achieving even weak differential privacy

Claim 2.5. [14] *If a distribution ensemble* $\mathcal{D} = \mathcal{D}_d$ *on* $n(d)$*-row databases is* $(\alpha, T_{\mathrm{San}}, \mathcal{C})$*-hard-to-sanitize, then for every constant* $a > 0$ *and every* $\beta = \beta(d) \le 1 - 1/\mathrm{poly}(d)$*, no* $T_{\mathrm{San}}(d)$*-time sanitizer that is* (α, β)*-accurate with respect to* $\mathcal{C}$ *can achieve* $(a\log(n), (1-8\beta)/2n^{1+a})$*-differential privacy.*

In particular, for all constants $\varepsilon, \beta > 0$*, no* $T_{\mathrm{San}}(d)$*-time sanitizer can be simultaneously* (α, β)*-accurate and* $(\varepsilon, \mathrm{negl}(n))$*-differentially private.*

We could use a weaker definition of hard-to-sanitize distributions, which would still suffice to rule out differential privacy, and only require that for every efficient $\mathcal{A}$, there exists an adversary $\mathcal{T}_{\mathcal{A}}$ that almost always extracts a row of D from every α-accurate output of $\mathcal{A}(D)$. In our definition, we require that there exists a fixed adversary $\mathcal{T}$ that almost always extracts a row of D from every α-accurate output of any efficient $\mathcal{A}$. Reversing the quantifiers in this fashion only makes our negative results stronger.

In this paper, we are concerned with sanitizers that output synthetic databases, so we will relax Definition 2.4 by restricting the quantification over sanitizers to only those sanitizers that output synthetic data.

Definition 2.6. *(Hard-to-sanitize Distribution as Synthetic Data)* A database distribution ensemble $\mathcal{D}$ is $(\alpha, T_{\mathrm{San}}, \mathcal{C})$*-hard-to-sanitize as synthetic data* if the conditions of Definition 2.4 hold for every sanitizer $\mathcal{A}$ that outputs a synthetic database.

The definition of hard-to-sanitize databases can be specialized in a similar way to other output representations besides synthetic data (e.g., medians of synthetic databases).

3. Relationship with Hardness of Approximation

The objective of a privacy-preserving sanitizer is to reveal some properties of the underlying database without giving away enough information to reconstruct that database. This requirement has different implications for sanitizers that produce synthetic databases and those with arbitrary output.

The SuLQ framework of [6] is a well-studied and efficient technique for achieving (ε, δ)-differential privacy, with non-synthetic output. To get accurate, private output for a family of counting queries with predicates in $\mathcal{C}$, we can release a vector of noisy counts $(c(D) + Z_c)_{c\in\mathcal{C}}$ where the random variables $(Z_c)_{c\in\mathcal{C}}$ are drawn independently from a distribution suitable for preserving privacy (e.g., a Laplace distribution with standard deviation $O(|\mathcal{C}|/\varepsilon n)$ would suffice).

Consider the case of an n-row database D that contains satisfying assignments to a 3CNF formula φ, and suppose our concept class includes all disjunctions on three literals (or, equivalently, all conjunctions on three literals). Then, the technique above releases a set of noisy counts that describes a database in which every clause of φ is satisfied by most of the rows of D. However, sanitizers with synthetic-database output are required to produce a database that consists of rows that satisfy most of the clauses of φ.

Because of the noise added to the output, the requirement of a synthetic database does not strictly force the sanitizer to find a satisfying assignment for the given 3CNF. However, it is known to be NP-hard to find even approximate satisfying assignments to 3CNF formulae. In our main result, Theorem 4.4, we will show that there exists a distribution over databases that is hard-to-sanitize with respect to synthetic data for any concept class that is sufficient to express a hard-to-approximate constraint satisfaction problem.

3.1. *Hard to Approximate CSPs*

We define a *constraint satisfaction problem* to be the following.

Definition 3.1. *(Constraint Satisfaction Problem* (CSP)) For a non-decreasing function $q = q(d) \leq d$, a *family of* $q(d)$-CSP*s*, denoted $\Gamma = (\Gamma_d)_{d \in \mathbb{N}}$, is a sequence of sets Γ_d of Boolean predicates defined on up to $q(d)$ variables. We assume for convenience that $\Gamma_1 \subseteq \Gamma_2 \subseteq \ldots$. If $q(d)$ and Γ_d do not depend on d, then we refer to Γ as a *fixed family of* q-CSP*s*.

For every $d \geq q(d)$, let $\mathcal{C}_\Gamma^{(d)}$ be the class consisting of all predicates $c : \{0,1\}^d \to \mathbb{R}$ of the form $c(u_1, \ldots, u_d) = \gamma(u_{i_1}, \ldots, u_{i_{q(d)}})$ for some $\gamma \in \Gamma_d$ and $i_1, \ldots, i_{q(d)} \in [d]$. We call $\mathcal{C}_\Gamma = \cup_{d=0}^{\infty} \mathcal{C}_\Gamma^{(d)}$ the *class of constraints of* Γ. Finally, we say a multiset $\varphi \subseteq \mathcal{C}_\Gamma^{(d)}$ is a *d-variable instance of* $\mathcal{C}_\Gamma$ and each $\varphi_i \in \varphi$ is a *constraint of* φ.

We say that an assignment π *satisfies* the constraint φ_i if $\varphi_i(\pi) = 1$. For $\varphi = \{\varphi_1, \ldots, \varphi_m\}$, define

$$\mathrm{val}(\varphi, \pi) = \frac{1}{m} \sum_{i=1}^{m} \varphi_i(\pi) \qquad \text{and} \qquad \mathrm{val}(\varphi) = \max_{\pi \in \{0,1\}^d} \mathrm{val}(\varphi, \pi).$$

Our hardness results will apply to concept classes $\mathcal{C}_\Gamma^{(d)}$ for CSP families Γ with certain additional properties. Specifically we define,

Definition 3.2. *(Nice* CSP*)* A family $\Gamma = (\Gamma_d)_{d \in \mathbb{N}}$ of $q(d)$-CSPs *nice* if

1. $q(d) = d^{1-\Omega(1)}$, and
2. for every $d \in \mathbb{N}$, there exists a non-constant predicate $\varphi^* : \{0,1\}^{q(d)} \to \{0,1\}$, and two assignments $u_0^*, u_1^* \in \{0,1\}^{q(d)}$ such that $\varphi^*(u_0) = 0$ and $\varphi^*(u_1) = 1$ can be found in time poly(d).

We note that any fixed family of q-CSP that contains a non-constant predicate is a nice CSP. Indeed, these CSPs (e.g., conjunctions of 2 literals) are the main application of interest for our results. However, it will sometimes be useful to work with generalizations to nice CSPs with predicates of non-constant arity.

For our hardness result, we will need to consider a strong notion of hard constraint satisfaction problems, which is related to probabilistically checkable proofs. First, we recall the standard notion of hardness of approximation under Karp reductions (stated for additive, rather than multiplicative approximation error).

Definition 3.3. *(Inapproximability under Karp Reductions)* Let $\alpha, \gamma : \mathbb{N} \to [0,1]$ be functions. A family of CSPs $\Gamma = (\Gamma_d)_{d \in \mathbb{N}}$ is (α, γ)*-hard-to-approximate under Karp reductions* if there exists a polynomial-time computable function R such that for every circuit C with $|C|$ gates and input size $\overline{d}$, if we set $\varphi_C = R(C) \subseteq \mathcal{C}_\Gamma$, then

1. if C is satisfiable, then $\mathrm{val}(\varphi_C) \geq \gamma(d)$, and
2. if C is unsatisfiable, then $\mathrm{val}(\varphi_C) < \gamma(d) - \alpha(d)$.

For our hardness result, we will need a stronger notion of inapproximability, which says that we can efficiently transform satisfying assignments of C into solutions to φ_C of high value, and vice versa. In order to make the statement of our hardness result more precise, we also want to put explicit bounds on both the input length of the instance produced by the reduction and the time required to transform assignments to C into solutions to φ_C and vice versa.

Definition 3.4. *(Inapproximability under Levin Reductions)* Let $\alpha, \gamma : \mathbb{N} \to [0, 1]$ and $L : \mathbb{N} \to \mathbb{N}$ be functions. A family of CSPs $\Gamma = (\Gamma_d)_{d \in \mathbb{N}}$ is *(α, γ)-hard-to-approximate under Levin reductions with length blowup L* if there exist polynomial-time computable functions $R, \mathit{Enc}, \mathit{Dec}$ such that for every circuit C with $|C|$ gates and input of size $\overline{d}$

1. $\varphi_C = R(C) \subseteq \mathcal{C}_\Gamma^{(d)}$ for $d \le L(|C|)$, and $|R(C)| \le L(|C|)$,
2. for every $u \in \{0, 1\}^{\overline{d}}$ such that $C(u) = 1$, $\mathrm{val}(\varphi_C, \mathit{Enc}(u, C)) \ge \gamma(d)$,
3. for every $\pi \in \{0, 1\}^d$ such that $\mathrm{val}(\varphi_C, \pi) \ge \gamma(d) - \alpha(d)$, $C(\mathit{Dec}(\pi, C)) = 1$,
4. for every $u \in \{0, 1\}^{\overline{d}}$, $\mathit{Dec}(\mathit{Enc}(u, C)) = u$, and
5. Enc and Dec are both computable in time $\mathrm{poly}(|C|, d)$.

When we do not wish to specify the value γ, we will simply say that the family Γ is *α-hard-to-approximate under Levin reductions with length blowup L* to indicate that there exists such a $\gamma \in (\alpha, 1]$. When we do not specify a length blowup L, then L is assumed to be a polynomial. If we drop the requirement that R is efficiently computable, then we say that Γ is *(α, γ)-hard-to-approximate under inefficient Levin reductions with length blowup L*. Finally, if $L(s) \le s^{1+o(1)}$ then we will write that Γ is *(α, γ)-hard-to-approximate (or simply α-hard-to-approximate) under Levin reductions with almost linear length blowup*.

The notation $\mathit{Enc}, \mathit{Dec}$ reflects the fact that we think of the set of assignments π such that $\mathrm{val}(\varphi_C, \pi) \ge \gamma$ as a sort of error-correcting code on the satisfying assignments to C. Any π with value close to γ can be decoded to a valid satisfying assignment.

Levin reductions are a stronger notion of reduction than Karp reductions. To see this, let Γ be α-hard-to-approximate under Levin reductions, and let $R, \mathit{Enc}, \mathit{Dec}$ be the functions described in Definition 3.4. We now argue that for every circuit C, the formula $\varphi_C = R(C)$ satisfies conditions 1 and 2 of Definition 3.3. Specifically, if there exists an assignment $u \in \{0, 1\}^{\overline{d}}$ that satisfies C, then $\mathit{Enc}(u, C)$ satisfies at least a γ fraction of the constraints of φ_C. Conversely, if any assignment $\pi \in \{0, 1\}^d$ satisfies at least a $\gamma - \alpha$ fraction of the constraints of φ_C, then $\mathit{Dec}(\pi, C)$ is a satisfying assignment of C.

Variants of the PCP theorem can be used to show that essentially every class of CSP is hard-to-approximate in this sense. Indeed, $\pi = \mathit{Enc}(u, C)$ corresponds to a "probabilistically checkable proof" (PCP) that C is satisfiable: If we check a random constraint of $\varphi_C = R(C)$ (which requires reading only a few bits of π), then we will accept with probability at least γ. Conversely, if the above test passes for some string $\pi \in \{0, 1\}^d$ with probability at least $\gamma - \alpha$, then C must be satisfiable (as $\mathit{Dec}(\pi, C)$ is a satisfying assignment to C).

The fact that standard PCPs actually yield Levin reductions has been explicitly discussed and formalized by Barak and Goldreich [4] in the terminology of PCPs rather than reductions. (The function Enc is called "relatively efficient oracle-construction," and the function Dec is called "a proof-of-knowledge property.") They verify that these properties hold for the PCP construction of Babai et al. [2], whereas we need it for PCPs of constant query complexity.

While the properties probably hold for most (if not all) existing PCP constructions, the existence of the efficient "decoding" function *Dec* requires some verification. Here, we sketch a proof of the existence of PCPs of almost linear length blowup, as such PCPs allow us to obtain tighter reductions.

Theorem 3.5. *There exists a fixed family of constant-arity CSPs* Γ *that is* $1/2$*-hard-to-approximate under Levin reductions with almost linear length blowup.*

Proof sketch. At a high level, our PCP for the statement $C(u) = 1$ will consist of an encoding w of u under an efficient error correcting code and a "PCP of Proximity" [5,11] that w is an encoding of a satisfying assignment to C. To specify the construction concretely and verify its properties, we will use the following two objects:

1. An error-correcting code that corrects a constant fraction of errors and can be encoded and decoded in almost linear time. Specifically, a pair of algorithms $(\mathrm{Enc}^{\mathrm{ECC}}, \mathrm{Dec}^{\mathrm{ECC}})$ with the following properties: $\mathrm{Enc}^{\mathrm{ECC}}$ takes a message $u \in \{0,1\}^{\overline{d}}$, runs in time $\overline{d}^{1+o(1)}$, and outputs a codeword $w \in \{0,1\}^{d_{\mathrm{ECC}}}$ for $d_{\mathrm{ECC}} = O(\overline{d})$. $\mathrm{Dec}^{\mathrm{ECC}}$ that takes a string $\tilde{w} \in \{0,1\}^{d_{\mathrm{ECC}}}$, runs in time $(d_{\mathrm{ECC}})^{1+o(1)}$, and for some constant $\varepsilon > 0$, if $\tilde{w}$ is ε-close in relative Hamming distance to an encoding of some (unique) string $u \in \{0,1\}^{\overline{d}}$, then $\mathrm{Dec}^{\mathrm{ECC}}(\tilde{w}) = u$. Binary error-correcting codes achieving these parameters are known to exist. One can be obtained, for example, by concatenating certain non-binary Reed–Solomon codes, which can be encoded and decoded in almost linear time ([24], see also the discussion in [36]) with an asymptotically good binary error-correcting code that allows for polynomial time encoding and decoding.
2. A PCP of proximity (PCPP) with almost linear length, constant soundness, and constant arity where proofs can be encoded (but not necessarily decoded) efficiently. Specifically, the following pair of algorithms $(R^{\mathrm{PCPP}}, \mathrm{Enc}^{\mathrm{PCPP}})$. R^{PCPP} takes as input a circuit $C' : \{0,1\}^{d_{\mathrm{ECC}}} \to \{0,1\}$ and for some $d_{\mathrm{PCPP}} = |C'|^{1+o(1)}$, R^{PCPP} outputs an instance $\varphi_{C'}$ of a CSP with constant arity over $d_{\mathrm{ECC}} + d_{\mathrm{PCPP}}$ binary variables with at most $|C'|^{1+o(1)}$ clauses. $\varphi_{C'}$ will be such that:
   (a) If $C'(w) = 1$, then there exists a "proof" $\pi \in \{0,1\}^{d_{\mathrm{PCPP}}}$ such that $\mathrm{val}(\varphi_{C'}, (w,\pi)) = 1$. Furthermore $\mathrm{Enc}^{\mathrm{PCPP}}$ takes as input C' and $w \in \{0,1\}^{d_{\mathrm{ECC}}}$ such that $C'(w) = 1$, runs in time $\mathrm{poly}(|C'|, d_{\mathrm{ECC}})$, and outputs a proof $\pi \in \{0,1\}^{d_{\mathrm{PCPP}}}$ such that if $C'(w) = 1$ then $\mathrm{val}(\varphi_{C'}, (w,\pi)) = 1$. Ben-Sasson et al. [5, Theorem 3.3] showed how to construct PCPPs achieving these parameters.[3]
   (b) For some constant $\alpha > 0$, if $\tilde{w} \in \{0,1\}^{d_{\mathrm{ECC}}}$ has relative Hamming distance greater than ε from every satisfying assignment to C', then for every $\pi \in \{0,1\}^{d_{\mathrm{PCPP}}}$, it holds that $\mathrm{val}(\varphi_{C'}, (\tilde{w},\pi)) < 1 - \alpha$.

Given these two objects, we can obtain a fixed family of constant-arity CSPs that is α-hard-to-approximate under Levin reductions with almost linear length blowup as follows. From this, we can obtain a (different) fixed family of (larger) constant-arity

[3] This result also does not explicitly state the existence of the efficient encoder, but the proof of the completeness property gives an efficient algorithm to generate the correct proof π.

CSPs that is 1/2-hard-to-approximate under Levin reductions with almost linear length blowup. We will provide a proof of this fact in Theorem 3.7.

Given a circuit $C : \{0,1\}^{\overline{d}} \to \{0,1\}$, define the circuit $C' : \{0,1\}^{d_{\mathrm{ECC}}} \to \{0,1\}$ as follows: $C'(w) = 1$ if and only if $C(\mathrm{Dec}^{\mathrm{ECC}}(w)) = 1$ and $w = \mathrm{Enc}^{\mathrm{ECC}}(\mathrm{Dec}^{\mathrm{ECC}}(w))$. C' checks that w is a codeword of the error correcting code, and encodes a satisfying assignment to C. Observe that, since $\mathrm{Enc}^{\mathrm{ECC}}$ and $\mathrm{Dec}^{\mathrm{ECC}}$ are computable in almost linear time, $|C'| = |C|^{1+o(1)}$.

Now, we define the promised PCP reduction $(R, \mathit{Enc}, \mathit{Dec})$. Let $R(C) = R^{\mathrm{PCPP}}(C')$, let $\mathit{Enc}(C, u) = (w, \mathrm{Enc}^{\mathrm{PCPP}}(C', w))$ where $w = \mathrm{Enc}^{\mathrm{ECC}}(u)$, and let $\mathit{Dec}(\tilde{w}, \pi) = \mathrm{Dec}^{\mathrm{ECC}}(\tilde{w})$.

First, observe that the length of the proof is $d_{\mathrm{ECC}} + |C'|^{1+o(1)} = O(\overline{d}) + |C|^{1+o(1)}$, and the number of clauses is $|C'|^{1+o(1)} = |C|^{1+o(1)}$, so it satisfies the almost linear length blowup condition. Since $\mathrm{Enc}^{\mathrm{ECC}}$, $\mathrm{Dec}^{\mathrm{ECC}}$, and $\mathrm{Enc}^{\mathrm{PCPP}}$ are all computationally efficient, so are *Enc* and *Dec*. Also, if $C(u) = 1$, then the encoding of u satisfies every constraint of φ_C. Specifically, by (a), if $C(u) = 1$, and $(w, \pi) = \mathit{Enc}(C, u)$, then $\mathrm{val}(\varphi_C, (w, \pi)) = 1$.

All that remains to be verified is that any "proof" π that satisfies sufficiently many constraints in φ_C can be efficiently decoded to find a satisfying assignment to C. That is, if $\mathrm{val}(\varphi_C, (\tilde{w}, \pi)) \geq 1 - \alpha$, then $C(\mathit{Dec}(\tilde{w}, \pi)) = 1$. If $\mathrm{val}(\varphi_C, (\tilde{w}, \pi)) \geq 1 - \alpha$, then, by (b), there exists w' such that $\tilde{w}$ and w' are ε-close in relative Hamming distance and $C'(w') = 1$. Because $C'(w') = 1$, w' must be a codeword in the error correcting code, so there exists $u \in \{0,1\}^{\overline{d}}$ such that $w' = \mathrm{Enc}^{\mathrm{ECC}}(u)$. Also, $C(\mathrm{Dec}^{\mathrm{ECC}}(w')) = C(u) = 1$. Since $w' = \mathrm{Enc}^{\mathrm{ECC}}(u)$ and $\tilde{w}$ is ε-close to w' in relative Hamming distance, $\mathrm{Dec}^{\mathrm{ECC}}(\tilde{w}) = u$. Therefore, $C(\mathrm{Dec}^{\mathrm{ECC}}(\tilde{w})) = 1$, as desired. □

It seems likely that optimized PCP/inapproximability results (like [23]) are also Levin reductions, which would yield fairly large values for α for natural CSPs (e.g., $\alpha = 1/8 - \varepsilon$ if Γ contains all conjunctions of 3-literals, because then $\mathcal{C}_\Gamma$ contains MAX 3-SAT.)

Now, we state a general result for all families of CSPs that are closed under negation.

Theorem 3.6. *For every fixed family of CSPs Γ that is closed under negation and contains a function that depends on at least two variables, there is a constant $\alpha = \alpha(\Gamma) > 0$ such that Γ is α-hard to approximate under Levin reductions with almost linear length blowup.*

Proof sketch. Hardness of approximation under Karp reductions follows directly from the classification theorems of Creignou [9] and Khanna et al. [26]. These theorems show that all CSPs are either α-hard under Karp reductions for some constant $\alpha > 0$ or can be solved optimally in polynomial time. By inspection, the only CSPs that fall into the polynomial-time cases (0-valid, 1-valid, and 2-monotone) and are closed under negation are those containing only dictatorships and constant functions.

To obtain hardness under Levin reductions, we can start with Theorem 3.5 and apply the same series of reductions that are used in the works above. First, we can apply a series of approximation-preserving reductions from arbitrary CSPs to MAX-CUT [31], which can be verified to preserve efficiency of decoding. (Indeed, the correctness of the reductions is proven by specifying how to encode and decode.) We can then apply the reduction from MAX-CUT to any other CSP that is closed under negation from [9,26].

These reductions all involve constant-sized *gadgets* that allow encoding and decoding to be done locally and very efficiently and preserve almost linear length blowup. Roughly, given an instance ϕ of a CSP Γ with d variables and m clauses, we can obtain an instance ϕ' of a CSP Γ' as follows: For each clause $\phi_i \in \phi$, there will be a constant number of clauses in ϕ', and these clauses will be over the original variables of ϕ_i plus some constant number of auxiliary variables that are specific to ϕ'. Since we add a constant number of variable per clause, the total number of variables in ϕ' is $d + O(m)$, and since ϕ' contains a constant number of clauses per clause of ϕ, the total number of clauses is $O(m)$; thus, the reduction preserves almost linear length blowup.

The analysis shows that if an assignment π to the variables of ϕ satisfies the clause ϕ_i, then there is a way to set the auxiliary variables to satisfy a large fraction of the corresponding clauses in ϕ', and since the number of auxiliary variables is constant, the transformation preserves efficient encoding. Conversely, if an assignment π' to ϕ' satisfies a large fraction of the clauses corresponding to a given clause ϕ_i, then the non-auxiliary variables of π' must satisfy the clause ϕ_i. Therefore, if π' satisfies a large enough fraction of ϕ', then there is a subassignment π that satisfies a large enough fraction of ϕ, to which we can apply the decoder for ϕ. Therefore, the transformation also preserves efficient decoding.

The precise values of the unspecified fractions of clauses will determine the new factor α for the target CSP family Γ'. See e.g., [26, Section 3] for a more detailed definition of a gadget (called an *implementation* in [26]) and how they allow efficient encoding and decoding. □

For some of our results, we will need CSPs that are very hard to approximate (under possibly inefficient reductions), which we can obtain by "sequential repetition" of constant-error PCPs.

Theorem 3.7. *There is a constant C such that for every $\varepsilon = \varepsilon(d) > 0$, the constraint family $\Gamma = (\Gamma_d)_{d\in\mathbb{N}}$ of $k(d)$-clause 3-CNF formulas is $(1-\varepsilon(d))$-hard-to-approximate under inefficient Levin reductions, for $k(d) = C\log(1/\varepsilon(d))$.*

Proof sketch. By Theorem 3.6, disjunctions of 3 literals (aka MAX-3CNF) are $(1-\delta, 1)$-hard-to-approximate under Levin reductions for some constant $\delta > 0$.[4] By taking $\ell(d) = \log_\delta(\varepsilon(d))$ sequential repetitions of this PCP, we get a PCP with completeness 1 and soundness $\varepsilon(d)$ whose constraints are 3-CNF formulas with $\ell(d) = \log_\delta(1/\varepsilon(d))$ clauses. Note that the resulting CSP will have arity at most $k(d) = 3\ell(d)$.

We have to check that this resulting PCP preserves the properties of inefficient Levin reductions. The encoder for the ℓ-fold sequential repetition is unchanged. If the initial reduction is $R(C) = \varphi_C = \{\varphi_1, \ldots, \varphi_m\}$ (a set of 3-literal disjunctions), then the reduction $R^\ell(C)$ for the ℓ-fold sequential repetition will produce m^ℓ, ℓ-clause 3-CNF formulae by

[4] Theorem 3.6 guarantees only $(\gamma-\delta, \gamma)$-hard-to-approximate, but it can be verified that the corresponding PCPs have "perfect completeness" so disjunctions of 3 literals are $(1-\delta, 1)$-hard-to-approximate.

taking every subcollection of ℓ clauses in φ_C. Specifically, for every $i_1, i_2, \ldots, i_\ell \in [m]$, $R^k(C)$ will contain a ℓ-clause 3-CNF formula $\varphi_{i_1} \wedge \varphi_{i_2} \wedge \cdots \wedge \varphi_{i_\ell}$.

The decoder also remains unchanged. If the value of an assignment π is at least δ^ℓ with respect to $R^\ell(C)$ then it must have value at least δ with respect to $R(C)$ and thus $Dec(\pi, C)$ will return a satisfying assignment to C, that is $C(Dec(\pi, C)) = 1$.

Notice that when $k = k(d) = \omega(1)$, the reduction will produce $m^{\omega(1)}$ clauses and be inefficient. Thus, we will have an inefficient Levin reduction if we want to obtain $\varepsilon(d) = o(1)$ from this construction. □

4. Hard-to-Sanitize Distributions from Hard CSPs

In this section, we prove that to efficiently produce a synthetic database that is accurate for the constraints of a CSP that is hard-to-approximate under Levin reductions, we must pay constant error in the worst case. Following [14], we start with a digital signature scheme, and a database of valid message–signature pairs. There is a verifying circuit C_{vk}, and valid message–signature pairs are satisfying assignments to that circuit. Now, we encode each row of database using the function *Enc*, described in Definition 3.4, that maps satisfying assignments to C_{vk} to assignments of the CSP instance $\varphi_{C_{vk}} = R(C_{vk})$ with value at least γ. Then, any assignment to the CSP instance that satisfies a $\gamma - \alpha$ fraction of clauses can be decoded to a valid message–signature pair. The database of encoded message–signature pairs is what we will use as our hard-to-sanitize distribution.

4.1. *Super-Secure Digital Signature Schemes*

Before proving our main result, we will formally define a *super-secure digital signature scheme*. These digital signature schemes have the property that it is infeasible under chosen-message attack to find a new message–signature pair that is different from all those obtained during the attack, even a new signature for an old message. First, we formally define digital signature schemes

Definition 4.1. *(Digital signature scheme)* For functions $L_{\mathrm{vk}}, T_{\mathrm{Ver}} : \mathbb{N} \to \mathbb{N}$. A $(L_{\mathrm{vk}}, T_{\mathrm{Ver}})$*-digital signature scheme with verification key length* L_{vk} *and verification time* T_{Ver} is a tuple of three probabilistic polynomial time algorithms $\Pi = (Gen, Sign, Ver)$ such that

1. *Gen* takes as input the security parameter 1^κ and outputs a key pair $(sk, vk) \leftarrow_{\mathrm{R}} Gen(1^\kappa)$ such that $vk \in \{0,1\}^{L_{\mathrm{vk}}(\kappa)}$ for a polynomial $L_{\mathrm{vk}}(\kappa)$.
2. *Sign* takes sk and a message $m \in \{0,1\}^*$ as input and outputs $\sigma \leftarrow_{\mathrm{R}} Sign_{sk}(m)$ such that $\sigma \in \{0,1\}^*$.
3. *Ver* takes vk and pair (m, σ) and deterministically outputs a bit $b \in \{0,1\}$, such that for every (sk, vk) in the range of *Gen*, and every message m, we have $Ver_{vk}(m, Sign_{sk}(m)) = 1$. Moreover, when $m \in \{0,1\}^\kappa$, and m is signed under (sk, vk) generated by $Gen(1^\kappa)$, then Ver_{vk} can be computed by a circuit of size $T_{\mathrm{Ver}}(\kappa)$ (for a polynomial T_{Ver}).

We define the security of a digital signature scheme with respect to the following game.

Definition 4.2. *(Weak forgery game)* For any signature scheme $\Pi = (\mathit{Gen}, \mathit{Sign}, \mathit{Ver})$ and probabilistic polynomial time adversary $\mathcal{F}$, $\texttt{WeakForge}(\mathcal{F}, \Pi, \kappa, Q_{\text{For}})$ is the following probabilistic experiment.

1. $(sk, vk) \leftarrow_{\text{R}} \mathit{Gen}(1^{\kappa})$.
2. $\mathcal{F}$ is given vk and oracle access to Sign_{sk}. The adversary adaptively queries Sign_{sk} on a set of at most Q_{For} messages $M \subset \{0, 1\}^*$, receives a set of message–signature pairs $A \subset \{0, 1\}^*$, and outputs (m^*, σ^*).
3. The output of the game is 1 if and only if (1) $\mathit{Ver}_{vk}(m^*, \sigma^*) = 1$, and (2) $(m^*, \sigma^*) \notin A$.

The weak forgery game is easier for the adversary to win than the standard forgery game because the final condition requires that the signature output by $\mathcal{F}$ be different from all pairs $(m, \sigma) \in A$, but allows for the possibility that $m^* \in M$. In the standard definition, the final condition would be replaced by $m^* \notin M$. Thus, the adversary has more possible outputs that would result in a "win" under this definition than under the standard definition.

Definition 4.3. *(Super-secure digital signature scheme)* For functions $T_{\text{For}}, Q_{\text{For}} : \mathbb{N} \to \mathbb{N}$. A digital signature scheme $\Pi = (\mathit{Gen}, \mathit{Sign}, \mathit{Ver})$, is a $(T_{\text{For}}, Q_{\text{For}})$*-super-secure digital signature scheme (under adaptive chosen message attack)* if for every T_{For}-time adversary, $\mathcal{F}$,

$$\Pr[\texttt{WeakForge}(\mathcal{F}, \Pi, \kappa, Q_{\text{For}}(\kappa)) = 1] \leq \text{negl}(\kappa).$$

Although the above definition is stronger than the usual definition of existentially unforgeable digital signatures, in [21] it is shown how to modify known constructions [30,34] to obtain a super-secure digital signature scheme from any one-way function.

In our terminology, the existence of one-way functions implies the existence of a digital signature scheme that is a $(T_{\text{For}}, Q_{\text{For}})$-super-secure digital signature scheme for every polynomial $T_{\text{For}}, Q_{\text{For}}$ with verification key length $\text{poly}(\kappa)$ and verification time $\text{poly}(\kappa)$. Under stronger hardness assumptions, super-secure digital signature schemes with even better parameters exist. In particular, under certain hardness assumptions in ideal lattices, there exists a digital signature scheme secure against $T_{\text{For}} = 2^{\kappa^{1-o(1)}}$ time adversaries making any $Q_{\text{For}} = \text{poly}(\kappa)$ queries, and this signature scheme has verification key length $L_{\text{vk}} = \kappa^{1+o(1)}$ and verification time $T_{\text{Ver}} = \kappa^{1+o(1)}$ [28]. It is plausible that there exist super-secure digital signature schemes secure against $T_{\text{For}} = 2^{\kappa^{1-o(1)}}$ time adversaries making $Q_{\text{For}} = 2^{\kappa^{1-o(1)}}$ queries with almost linear verification key length and verification time, but we do not know of any candidate constructions.

4.2. *Hard-to-Sanitize Distributions*

We are now ready to construct a general form of database distribution ensemble, which we can instantiate with various CSPs and signature schemes to prove our hardness results.

Let $\Gamma = (\Gamma_d)_{d \in \mathbb{N}}$ be a family of nice $q(d)$-CSPs (Definition 3.2) and let $\mathcal{C}_\Gamma = \cup_{d=1}^{\infty} \mathcal{C}_\Gamma^{(d)}$ be the class of constraints of Γ (Definition 3.1). Assume that Γ is hard-to-approximate (for some parameters) under Levin reductions with length blowup L and let *Enc* be the promised encoder. Let $\Pi = (Gen, Sign, Ver)$ be a digital signature scheme with verification key length L_{vk} and verification time T_{Ver} and let C_{vk} be a circuit computing Ver_{vk}. Let $n : \mathbb{N} \to \mathbb{N}$ be a function.

We define the database distribution ensemble $\mathcal{D} = \mathcal{D}_d(n, \Gamma, \Pi, Enc)$ for any function $n : \mathbb{N} \to \mathbb{N}$. A sample from $\mathcal{D}_d$ consists of $n(d) + 1$ random message–signature pairs encoded as PCP witnesses with respect to the signature-verification algorithm. Each row will also contain an encoding of the verification key for the signature scheme using the non-constant constraint $\varphi^* : \{0,1\}^{q(d)} \to \{0,1\}$ in Γ_d and the assignments $u_0^*, u_1^* \in \{0,1\}^{q(d)}$ such that $\varphi^*(u_0^*) = 0$ and $\varphi^*(u_1^*) = 1$, as described in the definition of nice CSPs (Definition 3.2).

```
**Database Distribution Ensemble** $\mathcal{D} = \mathcal{D}_d(n, \Gamma, \Pi, Enc)$:
  Let $n = n(d)$. Let $\varphi^* : \{0,1\}^{q(d)} \to \{0,1\}$ be a non-constant constraint in $\Gamma_d$ and $u_0^*, u_1^* \in \{0,1\}^{q(d)}$
  be such that $\varphi^*(u_0^*) = 0$ and $\varphi^*(u_1^*) = 1$.
  Let $\kappa = \kappa(d)$ be the largest integer such that $L(T_{\mathrm{Ver}}(\kappa)) \le d/2$ and $L_{\mathrm{vk}}(\kappa) q(d) \le d/2$.
  $(sk, vk) \leftarrow_{\mathrm{R}} Gen(1^\kappa)$, let $vk = vk_1 vk_2 \ldots vk_\ell$, where $\ell = L_{\mathrm{vk}}(\kappa)$
  $(m_1, \ldots, m_{n+1}) \leftarrow_{\mathrm{R}} (\{0,1\}^\kappa)^{n+1}$
  for $i = 1$ to $n + 1$ do
    Let $y_i := Enc(m_i \| Sign_{sk}(m_i), C_{vk})$, be a PCP encoding of $m_i$ and its signature, padded with zeros
    to be of length exactly $d/2$
    Let $z_i := u_{vk_1}^* \| u_{vk_2}^* \| \ldots \| u_{vk_\ell}^*$, be an encoding of $vk$, padded with zeros to length exactly $d/2$
    Let $x_i := y_i \| z_i$ be the concatenation of these two strings
  end for
  return $D_0 := (x_1, \ldots, x_{n+1})$
```

Recall that $s_1 \| s_2$ denotes the concatenation of the strings s_1 and s_2. Before moving on to instantiating $\mathcal{D}$ and proving our hardness results, we make some observations about the construction. First, observe that the construction is well defined. That is, the length of y_i before padding is exactly $L(T_{\mathrm{Ver}}(\kappa))$, and the length of z_i before padding is exactly $L_{\mathrm{vk}}(\kappa) q(d)$ and κ was chosen so that these quantities are both at most $d/2$ and the above is well defined. Also, note that $\kappa(d) = d^{\Omega(1)}$. This statement holds because Γ is nice (Definition 3.2), so $q(d) = d^{1-\Omega(1)}$, and because L, T_{Ver}, and L_{vk} are all bounded by some polynomial in their input length. Finally, note that our distribution over d-column databases contains PCPs of length $L = L(T_{\mathrm{Ver}}(\kappa)) \le d/2$, thus $R(C_{vk}) \subseteq \mathcal{C}_\Gamma^{(d/2)} \subseteq \mathcal{C}_\Gamma^{(d)}$ (by Definition 3.1).

4.3. *Main Hardness Result*

We are now ready to state and prove our main hardness result.

Theorem 4.4. *Let $\Gamma = (\Gamma_d)_{d\in\mathbb{N}}$ be a family of nice (Definition 3.2) $q(d)$-CSPs such that $\Gamma_d \cup \neg\Gamma_d$ is α-hard-to-approximate under (possibly inefficient) Levin reductions with length blowup L, for $\alpha = \alpha(d) \in (0, 1/2)$, and let Enc be the promised encoder. Assume the existence of a $(T_{\text{For}}, Q_{\text{For}})$-super-secure digital signature scheme Π with verification key length L_{vk} and verification time T_{Ver}. $\kappa = \kappa(d)$ be the largest integer such that $L(T_{\text{Ver}}(\kappa)) \leq d/2$ and $L_{\text{vk}}(\kappa)q(d) \leq d/2$ as in the construction of $\mathcal{D}$. Let $T_{\text{San}}, n : \mathbb{N} \to \mathbb{N}$ be any functions such that for every $a > 0$*

$$n(d) \cdot d^a + T_{\text{San}}(d) = o(T_{\text{For}}(\kappa(d)))$$

and for every $d \in \mathbb{N}$

$$n(d) \leq Q_{\text{For}}(\kappa(d)).$$

Then, the distribution $\mathcal{D} = \mathcal{D}_d(n, \Gamma, \Pi, Enc)$ on $n(d)$-row databases is $(\alpha, T_{\text{San}}(d), \mathcal{C}_\Gamma^{(d)})$-hard-to-sanitize as synthetic data.

Proof. Let $\Pi = (Gen, Sign, Ver)$ be the assumed digital signature scheme, and let C_{vk} be a circuit computing Ver_{vk}. Let Γ be the assumed family of nice $q(d)$-CSPs that is α-hard-to-approximate under (possibly inefficient) Levin reductions with length blowup L. Let R, Enc, Dec be the functions corresponding to the Levin reduction to Γ and $\gamma = \gamma(d) \in (\alpha, 1]$ be the parameter from Definition 3.4.

Every valid pair $(m, Sign_{sk}(m))$ is a satisfying assignment of the circuit C_{vk}; hence, every row of D_0 constructed by $\mathcal{D}$ will satisfy at least a γ fraction of the clauses of the formula $\varphi_{C_{vk}} = R(C_{vk}) \subseteq \mathcal{C}_\Gamma^{(d)}$. Additionally, for every bit of the verification key, there is a block of $q = q(d)$ bits in each row that contains either a satisfying assignment or a non-satisfying assignment of φ^*, depending on whether that bit of the key is 1 or 0. Specifically, let

$$\varphi_j^*(x) = \varphi^*(x_{d/2+(j-1)q+1}, x_{d/2+(j-1)q+2}, \ldots, x_{d/2+jq})$$

for $j = 1, 2, \ldots, L_{\text{vk}}(\kappa)$. Then, by construction, $\varphi_j^*(D_0) = vk_j$, the jth bit of the verification key. Note that $\varphi_j^* \in \mathcal{C}_\Gamma^{(d)}$ for $j = 1, 2, \ldots, \ell$, by our construction of $\mathcal{C}_\Gamma^{(d)}$ (Definition 3.1). □

We now prove the following two lemmas that will establish $\mathcal{D}$ is hard-to-sanitize:

Lemma 4.5. *There exists a polynomial-time adversary $\mathcal{T}$ such that for every T_{San}-time sanitizer $\mathcal{A}$,*

$$\Pr_{\substack{(D,D',i)\leftarrow_R \tilde{\mathcal{D}} \\ \mathcal{A}'s \text{ and } \mathcal{T}'s \text{ coins}}} \left[(\mathcal{A}(D) \textit{ is } \alpha\textit{-accurate for } \mathcal{C}_\Gamma^{(d)}) \wedge (\mathcal{T}(\mathcal{A}(D)) \cap D = \emptyset)\right] \leq \text{negl}(d) \tag{1}$$

Proof. Our privacy adversary tries to find a row of the original database by trying to PCP-decode each row of the "sanitized" database and then re-encoding it. In order to do so, the adversary needs to know the verification key used in the construction of the database, which it can discover from the answers to the queries φ_j^*, defined above. Formally, we define the privacy adversary by means of a subroutine that tries to learn the verification key and then PCP-decode each row of the input database:

```
Subroutine $\mathcal{K}(\widehat{D})$:
  Let $d$ be the dimension of rows in $\widehat{D}$, let $\kappa$ be defined as in the construction of $\mathcal{D}$,
  and $\ell = L_{\mathrm{vk}}(\kappa)$.
  for $j = 1$ to $\ell$ do
    $\widehat{vk}_j = \left[\varphi_j^*(\widehat{D}) \text{ rounded to } \{0,1\}\right]$
  end for
  return $\widehat{vk}_1 \| \widehat{vk}_2 \| \ldots \| \widehat{vk}_\ell$

Subroutine $\mathcal{T}_0(\widehat{D})$:
  Let $\hat{n}$ be the number of rows in $\widehat{D}$, $\widehat{vk} = \mathcal{K}(\widehat{D})$
  for $i = 1$ to $\hat{n}$ do
    if $C_{\widehat{vk}}(Dec(\hat{x}_i, C_{\widehat{vk}})) = 1$ then
      return $Dec(\hat{x}_i, C_{\widehat{vk}})$
    end if
  end for
  return $\perp$

Privacy Adversary $\mathcal{T}(\widehat{D})$:
  Let $\widehat{vk} = \mathcal{K}(\widehat{D})$.
  return $Enc(\mathcal{T}_0(\widehat{D}), C_{\widehat{vk}})$
```

Let $\mathcal{A}$ be a T_{San}-time sanitizer, we will show that Inequality (1) holds.

Claim 4.6. *If $\widehat{D} = \mathcal{A}(D)$ is α-accurate for $\mathcal{C}_\Gamma^{(d)}$, then $\mathcal{T}_0(\widehat{D})$ outputs a pair (m, σ) s.t. $C_{vk}(m, \sigma) = 1$.*

Proof. First, we argue that if $\widehat{D}$ is α-accurate for $\mathcal{C}_\Gamma^{(d)}$ for $\alpha < 1/2$, then $\mathcal{K}(\widehat{D}) = vk$, where vk is the verification key used in the construction of D_0. By construction, $\varphi_j^*(D) = vk_j$. If $vk_j = 0$ and $\widehat{D}$ is α-accurate for D, then $\varphi_j^*(\widehat{D}) \leq \alpha < 1/2$, and $\widehat{vk}_j = vk_j$. Similarly, if $vk_j = 1$, then $\varphi_j^*(\widehat{D}) \geq 1 - \alpha > 1/2$, and $\widehat{vk}_j = vk_j$. Thus, for the rest of the proof we will be justified in substituting vk for $\widehat{vk}$.

Next, we show that if $\widehat{D}$ is α-accurate, then $\mathcal{T}_0(\widehat{D}) \neq \perp$. It is sufficient to show there exists $\hat{x}_i \in \widehat{D}$ such that $\mathrm{val}(\varphi_{C_{vk}}, x_i) \geq \gamma - \alpha$, which implies $C_{vk}(Dec(\hat{x}_i, C_{vk})) = 1$.

Since every $(m_i, Sign_{sk}(m_i))$ pair is a satisfying assignment to C_{vk}, the definition of Enc (Definition 3.4) implies that each row x_i of D has $\mathrm{val}(\varphi_{C_{vk}}, x_i) \geq \gamma$. Thus, if

$\varphi_{C_{vk}} = \{\varphi_1, \ldots, \varphi_m\}$, then

$$\frac{1}{m}\sum_{j=1}^{m} \varphi_j(D) = \frac{1}{m}\sum_{j=1}^{m}\left(\frac{1}{n}\sum_{i=1}^{n}\varphi_j(x_i)\right) = \frac{1}{n}\sum_{i=1}^{n}\mathrm{val}(\varphi_{C_{vk}}, x_i) \geq \gamma.$$

Since $\widehat{D}$ is α-accurate for $\mathcal{C}_{\Gamma}^{(d)}$, and for every constraint φ_j, either $\varphi_j \in \Gamma$ or $\neg\varphi_j \in \Gamma$, then for every constraint $\varphi_j \in \varphi_{C_{vk}}$, we have $\varphi_j(\widehat{D}) \geq \varphi_j(D) - \alpha$. Thus,

$$\frac{1}{\hat{n}}\sum_{i=1}^{\hat{n}}\mathrm{val}(\varphi_{C_{vk}}, \hat{x}_i) = \frac{1}{m}\sum_{j=1}^{m}\varphi_j(\widehat{D}) \geq \frac{1}{m}\sum_{j=1}^{m}\varphi_j(D) - \alpha \geq \gamma - \alpha.$$

So for at least one row $\hat{x} \in \widehat{D}$ it must be the case that $\mathrm{val}(\varphi_{C_{vk}}, \hat{x}) \geq \gamma - \alpha$. The definition of *Dec* (Definition 3.4) implies $C_{vk}(\mathit{Dec}(\hat{x}, C_{vk})) = 1$. □

Now notice that if $\mathcal{T}_0(\mathcal{A}(D))$ outputs a valid message–signature pair but $\mathcal{T}(\mathcal{A}(D)) \cap D = \emptyset$, then this means $\mathcal{T}_0(\mathcal{A}(D))$ is forging a new signature not among those used to generate D. Formally, we construct a signature forger as follows:

```
Forger $\mathcal{F}(vk)$ with oracle access to $\mathit{Sign}_{sk}$:
    Use the oracle $\mathit{Sign}_{sk}$ to generate an $n$-row database $D$ just as in the definition
    of $\mathcal{D}_d$ (consisting of PCP encodings of valid message–signature pairs and an
    encoding of $vk$).
    Let $\widehat{D} := \mathcal{A}(D)$
    return $\hat{x}^* := \mathcal{T}_0(\widehat{D})$
```

First, we analyze the running time of $\mathcal{F}$. In order to construct D, the forger must PCP-encode the signatures, which requires time $n(d) \cdot \mathrm{poly}(T_{\mathrm{Ver}}(\kappa)) = n(d) \cdot \mathrm{poly}(d)$. Running $\mathcal{A}$ requires time $T_{\mathrm{San}}(d)$ by assumption. Finally, the time to run $\mathcal{T}_0(\widehat{D})$ to decode a message–signature pair is $n(d) \cdot \mathrm{poly}(d)$, since $\mathcal{T}_0$ has to run the PCP decoder on each row of $\widehat{D}$. Put together, the running time of $\mathcal{F}$ is $n(d) \cdot \mathrm{poly}(d) + T_{\mathrm{San}}(d) = o(T_{\mathrm{For}}(\kappa(d)))$, where the inequality is an assumption of the theorem. Next, we note that the forger makes at most $n(d) \leq o(Q_{\mathrm{For}}(\kappa(d)))$ queries to Sign_{sk}, and this inequality is also an assumption of the theorem.

Now observe that running $\mathcal{F}$ in the experiment $\texttt{WeakForge}(\mathcal{F}, \Pi, \kappa, Q_{\mathrm{For}}(\kappa))$ is equivalent to running $\mathcal{T}$ in the experiment of Inequality (1), except that $\mathcal{F}$ does not re-encode the output of $\mathcal{T}_0(\mathcal{A}(D))$. By the super-security of the signature scheme, if the $\hat{x}^*$ output by $\mathcal{F}$ is a valid message–signature pair (as holds if $\mathcal{A}(D)$ is α-accurate for $\mathcal{C}_{\Gamma}^{(d)}$, by Claim 4.6), then it must be one of the message–signature pairs used to construct D (except with probability $\mathrm{negl}(\kappa) = \mathrm{negl}(d^{\Omega(1)}) = \mathrm{negl}(d)$). This implies that $\mathcal{T}(\mathcal{A}(D)) = \mathit{Enc}(\hat{x}^*, C_{vk}) \in D$ (except with negligible probability). Thus, we have

$$\Pr_{\substack{(D,D',i)\leftarrow_{\mathrm{R}}\tilde{\mathcal{D}} \\ \mathcal{A}'s \text{ coins}}}[\mathcal{A}(D) \text{ is } \alpha\text{-accurate for} \mathcal{C}_{\Gamma}^{(d)} \Rightarrow \mathcal{T}(\mathcal{A}(D)) \in D] \geq 1 - \mathrm{negl}(d),$$

which is equivalent to the statement of the lemma. □

Lemma 4.7.

$$\Pr_{\substack{(D,D',i)\leftarrow_R \tilde{\mathcal{D}}\\ \mathcal{A}'s \text{ and } \mathcal{T}'s \text{ coins}}}\left[\mathcal{T}(\mathcal{A}(D')) = x_i\right] \le \text{negl}(d)$$

Proof. Since the messages m_i used in D_0 are drawn independently, D' contains no information about the message m_i; thus, no adversary can, on input $\mathcal{A}(D')$, output the target row x_i except with probability $2^{-\kappa} = \text{negl}(d)$. □

These two claims suffice to establish that $\mathcal{D}$ is $(\alpha, \mathcal{C}_\Gamma)$-hard-to-sanitize as synthetic data. □

Theorem 1.1 in the introduction follows by combining Theorems 3.6 and 4.4.

Assuming the existence of an efficient super-secure digital signature scheme against almost-exponential-time adversaries, yields the following variants of Theorem 1.1.

Corollary 4.8. *Assume the existence of a digital signature scheme* Π *that is* $(T_{\text{For}}, Q_{\text{For}})$*-super-secure for* $T_{\text{For}} = 2^{\kappa^{1-o(1)}}$ *and* $Q_{\text{For}} = 2^{\kappa^{1-o(1)}}$*, and has verification key length* $L_{\text{vk}} = \kappa^{1+o(1)}$ *and verification time* $T_{\text{Ver}} = \kappa^{1+o(1)}$*. Then, there exists a family* Γ *of constant-arity CSPs such that for some* $n(d) = 2^{d^{1-o(1)}}$*, the distribution ensemble* $\mathcal{D} = \mathcal{D}_d(n, \Gamma, \Pi)$ *on* $n(d)$*-row databases is* $(1/2, T_{\text{San}}, \mathcal{C}_\Gamma^{(d)})$*-hard-to-sanitize as synthetic data for some* $T_{\text{San}}(d) = 2^{d^{1-o(1)}}$.

Proof. By Theorem 3.5, there exists a fixed family of q-CSPs that is 1/2-hard-to-approximate under Levin reductions with almost linear length blowup. That is, $L(s) = s^{1+o(1)}$. Now, in the construction of $\mathcal{D}$, we can choose $\kappa = d^{1-o(1)}$ such that $L(T_{\text{Ver}}(\kappa)) \le d/2$ and $L_{\text{vk}} \cdot q \le d/2$. Thus, we have $T_{\text{For}}(\kappa) = 2^{\kappa^{1-o(1)}} = 2^{d^{1-o(1)}}$.

Now, we can find some function $T_{\text{San}}(d) = 2^{d^{1-o(1)}}$ such that

$$n(d)\cdot d^a + T_{\text{San}}(d) = o(T_{\text{For}}(\kappa(d)))$$

for every $a > 0$. Additionally, we can find some function $n(d) = 2^{d^{1-o(1)}}$ such that

$$n(d) \le Q_{\text{For}}(\kappa(d)).$$

Thus, the assumptions of Theorem 4.4 are satisfied. □

In particular, Corollary 4.8 shows that if there is a signature scheme with the assumed parameters, then there is no algorithm that takes a database of size $n = 2^{d^{1-\Omega(1)}}$, runs in time $2^{d^{1-\Omega(1)}}$ and outputs a private synthetic database that is accurate for each of the 3^d marginal queries. (Recall that a marginal query computes the fraction of rows satisfying some Boolean conjunction.) To see that Corollary 4.8 applies to marginal queries, we note that if there is a fixed family of CSPs Γ of arity k that is 1/2-hard-to-approximate, then the family of CSPs Γ' consisting of only the conjunction of k literals is $1/2^{k+1}$-hard-to-approximate (under the same reduction). Thus, Corollary 4.8 applies to the family of queries $\mathcal{C}^{(d)}_{\Gamma'}$ consisting of all conjunctions of k-literals on $\{0, 1\}^d$, which is a subset of all marginal queries.

In contrast, subsequent work [37] shows that there is a differentially private algorithm that takes a database of size $n = 2^{\tilde{O}(\sqrt{d})}$, runs in time $2^{\tilde{O}(\sqrt{d})}$, and outputs a summary that is not a synthetic database, yet allows a data analyst to compute an accurate answer to any marginal query in time $2^{\tilde{O}(\sqrt{d})}$. Thus, [37] shows that, under the assumption in Corollary 4.8, differentially private algorithms that do not generate synthetic data can answer all marginal queries more efficiently than those that do generate synthetic data. Although it is plausible that a signature scheme satisfying the assumption in Corollary 4.8 exists, we are not aware of any candidates.

The algorithm of Blum et al. [7] requires exponential time, not just super-polynomial time (as shown necessary by Theorem 1.1), even for answering polynomial-sized families of queries. Specifically, for some polynomial $n(d)$, and every constant k, their algorithm takes a database with $n(d)$ rows and outputs a private synthetic database accurate for every k-way marginal query. (Recall that a k-way marginal query computes the fraction of rows satisfying some Boolean conjunction consisting of at most k literals, and there are at most $O(d)^k$ such queries on the data universe $\{0, 1\}^d$.) However, their algorithm requires time at least 2^d even for this special case. Subsequent work [8] showed how to achieve similar parameters to [7] for k-way marginal queries in subexponential time by avoiding synthetic data. Specifically, they gave a differentially private algorithm that, for some polynomial $n(d)$, and every constant k, takes a database with $n(d)$ rows, runs in time $2^{d^{1-\Omega(1)}}$, and outputs accurate answers to every k-way marginal query. We show that these parameters would be impossible for algorithms generating private synthetic data under an assumption weaker than that in Corollary 4.8.

Corollary 4.9. *Assume the existence of a digital signature scheme* Π *that is* $(T_{\mathrm{For}}, Q_{\mathrm{For}})$-*super-secure for* $T_{\mathrm{For}} = 2^{\kappa^{1-o(1)}}$ *and every polynomial* $Q_{\mathrm{For}} = Q_{\mathrm{For}}(\kappa)$*, and has verification key length* $L_{\mathrm{vk}} = \kappa^{1+o(1)}$ *and verification time* $T_{\mathrm{Ver}} = \kappa^{1+o(1)}$*. Then, there exists a family* Γ *of constant-arity CSPs such that for every polynomial* $n = n(d)$*, the distribution ensemble* $\mathcal{D} = \mathcal{D}_d(n, \Gamma, \Pi)$ *on* $n(d)$*-row databases is* $(1/2, T_{\mathrm{San}}, \mathcal{C}^{(d)}_{\Gamma})$-*hard-to-sanitize as synthetic data for some* $T_{\mathrm{San}}(d) = 2^{d^{1-o(1)}}$.

The proof of Corollary 4.9 closely follows that of Corollary 4.8. As with Corollary 4.8, Corollary 4.9 can be made to apply to k-way marginal queries. As we discussed in Sect. 4.1, a signature scheme as in the hypothesis of Corollary 4.9 exists under certain hardness assumptions in ideal lattices [28].

5. Relaxed Synthetic Data

The proof of Theorem 4.4 requires that the sanitizer outputs a synthetic database. In this section, we present similar hardness results for sanitizers that produce other forms of output, as long as they still produce a collection of elements from $\{0, 1\}^d$, that are interpreted as the data of (possibly "fake") individuals. More specifically, we consider sanitizers that output a database $\widehat{D} \in (\{0,1\}^d)^{\hat{n}}$ but are interpreted using an evaluation function of the following form: To evaluate predicate $c \in \mathcal{C}$ on $\widehat{D}$, apply c to each row $\hat{x}_i$ of $\widehat{D}$ to get a string of $\hat{n}$ bits, and then apply a function $f : \{0,1\}^{\hat{n}} \times \mathcal{C} \to [0,1]$ to determine the answer. For example, when the sanitizer outputs a synthetic database, we have $f(b_1, \ldots, b_{\hat{n}}, c) = (1/\hat{n}) \sum_{i=1}^{\hat{n}} b_i$, which is just the fraction of rows that get labeled with a 1 by the predicate c (independent of c).

We now give a formal definition of *relaxed synthetic data*

Definition 5.1. *(Relaxed Synthetic Data)* A sanitizer $\mathcal{A} : (\{0,1\}^d)^n \to (\{0,1\}^d)^{\hat{n}}$ with evaluator $\mathcal{E}$ outputs *relaxed synthetic data* for a family of predicates $\mathcal{C}$ if there exists $f : \{0,1\}^{\hat{n}} \times \mathcal{C} \to [0,1]$ such that

- For every $c \in \mathcal{C}$

$$\mathcal{E}(\widehat{D}, c) = f(c(\hat{x}_1), c(\hat{x}_2), \ldots, c(\hat{x}_{\hat{n}}), c),$$

  and
- f is monotone[5] in the first $\hat{n}$ inputs.

This relaxed notion of synthetic data is of interest because many natural approaches to sanitizing yield outputs of this type. In particular, several previous sanitization algorithms [7,18,35] produce a *set* of synthetic databases and answer a query by taking a median over the answers given by the individual databases. Sanitizers that use medians of synthetic databases no longer have the advantage that they are "interchangeable" with the original data, but are still be desirable for data releases because they retain the property that a small data structure can give accurate answers to a large number of queries. We view such databases as a single synthetic database but require that f have a special form. Unfortunately, the sanitizers of [35] and [18] run in time exponential in the dimension of the data, d, and the results of the next subsection show this limitation is inherent even for simple concept classes.

Throughout this section, we will continue to use $c(\widehat{D})$ to refer to the answer given by $\widehat{D}$ *when interpreted as a synthetic database*.

We now present our hardness results for relaxed synthetic data where the function f takes the median over synthetic database (Sect. 5.1), where f is an arbitrary monotone, symmetric function (Sect. 5.2), or when the family of concepts contains CSPs that are very hard to approximate (Sect. 5.3). Our proofs use the same construction of hard-to-sanitize databases as Theorem 4.4 with a modified analysis and parameters to show that the output must still contain a PCP-decodable row.

5.1. *Hardness of Sanitizing as Medians*

In this section, we establish that the distribution $\mathcal{D}$ used in the proof of Theorem 4.4 is hard-to-sanitize as medians of synthetic data, formally defined as:

[5] Given two vectors $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_n)$ we say $b \succeq a$ iff $b_i \geq a_i$ for every $i \in [n]$. We say a function $f : \{0,1\}^n \to [0,1]$ is *monotone* if $b \succeq a \Longrightarrow f(b) \geq f(a)$.

Definition 5.2. (Medians of Synthetic Data) A sanitizer $\mathcal{A} : (\{0,1\}^d)^n \to (\{0,1\}^d)^{\hat{n}}$ with evaluator $\mathcal{E}$ outputs *medians of synthetic data* if there is a partition $[\hat{n}] = S_1 \cup S_2 \cdots \cup S_\ell$ such that

$$\mathcal{E}(\hat{x}_1, \ldots, \hat{x}_{\hat{n}}, c) = \text{median}\left\{\frac{1}{|S_1|}\sum_{i \in S_1} c(\hat{x}_i), \frac{1}{|S_2|}\sum_{i \in S_2} c(\hat{x}_i), \ldots, \frac{1}{|S_\ell|}\sum_{i \in S_\ell} c(\hat{x}_i)\right\}.$$

Note that medians of synthetic data are a special case of relaxed synthetic data. In the following, we rule out efficient sanitizers with medians of synthetic data for CSPs that are hard to approximate within a multiplicative factor larger than 2. By Theorem 3.7, these CSPs include k-clause 3-CNF formulas for some constant k.

Theorem 5.3. *Let $\Gamma = (\Gamma_d)_{d\in\mathbb{N}}$ be a family of nice (Definition 3.2) $q(d)$-CSPs such that $\Gamma_d \cup \neg\Gamma_d$ is $((\alpha+\gamma)/2, \gamma)$-hard-to-approximate under (possibly inefficient) Levin reductions with length blowup L, for $\alpha = \alpha(d) \in (0, 1/2)$, and let Enc be the promised encoder. Assume the existence of a $(T_{\text{For}}, Q_{\text{For}})$-digital signature scheme Π with verification key length L_{vk} and verification time T_{Ver}. Let $\kappa = \kappa(d)$ be the largest integer such that $L(T_{\text{Ver}}(\kappa)) \le d/2$ and $L_{\text{vk}}(\kappa)q(d) \le d/2$ as in the construction of $\mathcal{D}$. Let $T_{\text{San}}, n : \mathbb{N} \to \mathbb{N}$ be any functions such that for every $a > 0$*

$$n(d) \cdot d^a + T_{\text{San}}(d) = o(T_{\text{For}}(\kappa(d)))$$

and for every $d \in \mathbb{N}$

$$n(d) \le Q_{\text{For}}(\kappa(d)).$$

Then, the distribution $\mathcal{D} = \mathcal{D}_d(n, \Gamma, \Pi, Enc)$ on $n(d)$-row databases is $(\alpha, T_{\text{San}}(d), \mathcal{C}_\Gamma^{(d)})$-hard-to-sanitize as medians of synthetic data.

Proof. Let Γ be the assumed family of nice $q(d) -$ CSPs that is $((\alpha+\gamma)/2, \gamma)$-hard-to-approximate under (possibly inefficient) Levin reductions with length blowup L. Let R, Enc, Dec be the functions corresponding to the Levin reduction to Γ. Let $\mathcal{A}(D) = \widehat{D}$ and let $\left\{\widehat{D}_1, \widehat{D}_2, \ldots, \widehat{D}_\ell\right\}$ be the partition of the rows of $\widehat{D}$ corresponding to $S_1, \ldots, S_\ell$, i.e., $\widehat{D}_i = (\hat{x}_j)_{j \in S_i}$.

Assuming that $\widehat{D}$ is α-accurate as medians of synthetic data, we will show that there must exist a row $\hat{x} \in \widehat{D}$ such that $\text{val}(\varphi_{C_{vk}}, \hat{x}) \ge \gamma - (\alpha+\gamma)/2 = (\gamma-\alpha)/2$. To do so, we observe that if $\widehat{D}$ is accurate as medians of synthetic databases, then for each predicate, half of $\widehat{D}$'s synthetic databases must give an answer that is "close to D's answer." Thus, one of these synthetic datsbases must be "close" to D for half of the predicates in $\varphi_{C_{vk}}$. By our construction of D, we conclude that each of these predicates is satisfied by many rows of this synthetic database and thus some row satisfies enough of the predicates to decode a message–signature pair.

We need to show that there exists an adversary $\mathcal{T}$ such that for every T_{San}-time sanitizer $\mathcal{A}$,

$$\Pr_{\substack{(D,D',i)\leftarrow_{\mathrm{R}}\tilde{\mathcal{D}}\\ \mathcal{A}'s \text{ and } \mathcal{T}'s \text{ coins}}}\left[(\mathcal{A}(D) \text{ is } \alpha\text{-accurate for } \mathcal{C}_{\Gamma}^{(d)}) \wedge (\mathcal{T}(\mathcal{A}(D)) \cap D = \emptyset)\right] \leq \mathrm{negl}(d) \tag{2}$$

To do so, we will use the same subroutine $\mathcal{T}_0(\widehat{D})$ we used for the proof of Lemma 4.5. That is, we consider a subroutine that looks for rows satisfying sufficiently many clauses of $\varphi_{C_{vk}}$ and returns the PCP-decoding of that row. It will suffice to establish the following claim, analogous to Claim 4.6:

Claim 5.4. *If $\widehat{D}$ is α-accurate for $\mathcal{C}_{\Gamma}^{(d)}$ as medians of synthetic data, then $\mathcal{T}_0(\widehat{D})$ outputs a pair (m, σ) s.t. $C_{vk}(m, \sigma) = 1$.*

Proof. As in the proof of Claim 4.6, if $\widehat{D}$ is α-accurate for $\mathcal{C}_{\Gamma}^{(d)}$ for $\alpha < 1/2$, then $\mathcal{K}(\widehat{D}) = vk$, the verification key used in the construction of D_0. For the rest of the proof, we will be justified in substituting vk for $\widehat{vk}$.

If $\varphi_{C_{vk}} = \{\varphi_1, \ldots, \varphi_m\}$, then $\frac{1}{m}\sum_{j=1}^{m}\varphi_j(D) \geq \gamma$. We say that $\widehat{D}_k$ is *good for* φ_j if $\varphi_j(\widehat{D}_k) \geq \varphi_j(D) - \alpha$. Since the median over $\left\{\widehat{D}_1, \widehat{D}_2, \ldots, \widehat{D}_\ell\right\}$ is α-accurate for every constraint $\varphi_j \in \varphi_{C_{vk}}$, we have

$$\Pr_{k\leftarrow_{\mathrm{R}}[\ell]}\left[\widehat{D}_k \text{ is good for } \varphi_j\right] \geq \frac{1}{2}$$

Then,

$$\begin{aligned}
\mathop{\mathbb{E}}_{k\leftarrow_{\mathrm{R}}[\ell]}\left[\frac{1}{|S_k|}\sum_{i\in|S_k|}\mathrm{val}(\varphi_{C_{vk}}, \hat{x}_i)\right] &= \mathop{\mathbb{E}}_{k\leftarrow_{\mathrm{R}}[\ell]}\left[\frac{1}{m}\sum_{j=1}^{m}\varphi_j(\widehat{D}_k)\right]\\
&= \frac{1}{m}\sum_{j=1}^{m}\mathop{\mathbb{E}}_{k\leftarrow_{\mathrm{R}}[\ell]}\left[\varphi_j(\widehat{D}_k)\right]\\
&\geq \frac{1}{m}\sum_{j=1}^{m}\left(\Pr_{k\leftarrow_{\mathrm{R}}[\ell]}\left[\widehat{D}_k \text{ is good for } \varphi_j\right]\cdot\left(\varphi_j(D)-\alpha\right)\right)\\
&\geq \frac{1}{m}\sum_{j=1}^{m}\left(\frac{1}{2}\cdot\left(\varphi_j(D)-\alpha\right)\right)\\
&\geq \frac{\gamma-\alpha}{2}
\end{aligned}$$

□

So for at least one row $\hat{x} \in \widehat{D}$ it must be the case that $\mathrm{val}(\varphi_{C_{vk}}, \hat{x}) \geq (\gamma - \alpha)/2$. Since the distribution $\mathcal{D}$ is unchanged, Lemma 4.7 still holds in this setting. Thus, we have established that $\mathcal{D}$ is $(\alpha(d), T_{\mathrm{San}}(d), \mathcal{C}_{\Gamma}^{(d)})$-hard-to-sanitize as medians of synthetic data. □

5.2. *Hardness of Sanitizing with Symmetric Evaluation Functions*

In this section, we establish the hardness of sanitization for relaxed synthetic data where the evaluator function is symmetric.

Definition 5.5. *(Symmetric Relaxed Synthetic Data)* A sanitizer $\mathcal{A} : (\{0, 1\}^d)^n \to (\{0, 1\}^d)^{\hat{n}}$ with evaluator $\mathcal{E}$ outputs *symmetric relaxed synthetic data* if there exists a monotone function $g : [0, 1] \to [0, 1]$ such that

$$\mathcal{E}(\hat{x}_1, \dots, \hat{x}_{\hat{n}}, c) = g\left(\frac{1}{\hat{n}} \sum_{i=1}^{\hat{n}} c(\hat{x}_i)\right).$$

Note that symmetric relaxed synthetic data are also a special case of relaxed synthetic data. Our definition of symmetric relaxed synthetic data is actually symmetric in two respects, because we require that g does not depend on the predicate c and also that g only depends on the fraction of rows that satisfy c. Similar to medians of synthetic data, we show that it is intractable to produce a sanitization as symmetric relaxed synthetic data that is accurate when the queries come from a CSP that is hard to approximate.

Theorem 5.6. *Let $\Gamma = (\Gamma_d)_{d \in \mathbb{N}}$ be a family of nice (Definition 3.2) $q(d)$-CSPs that is closed under complement ($\Gamma_d = \neg\Gamma_d$) and is $\alpha + 1/2$-hard-to-approximate under (possibly inefficient) Levin reductions with length blowup L, for $\alpha = \alpha(d) \in (0, 1/2)$, and let Enc be the promised encoder. Assume the existence of a $(T_{\mathrm{For}}, Q_{\mathrm{For}})$-digital signature scheme Π with verification key length L_{vk} and verification time T_{Ver}. Let $\kappa = \kappa(d)$ be the largest integer such that $L(T_{\mathrm{Ver}}(\kappa)) \le d/2$ and $L_{\mathrm{vk}}(\kappa)q(d) \le d/2$ as in the construction of $\mathcal{D}$. Let $T_{\mathrm{San}}, n : \mathbb{N} \to \mathbb{N}$ be any functions such that for every $a > 0$*

$$n(d) \cdot d^a + T_{\mathrm{San}}(d) = o(T_{\mathrm{For}}(\kappa(d)))$$

and for every $d \in \mathbb{N}$

$$n(d) \le Q_{\mathrm{For}}(\kappa(d)).$$

Then, the distribution $\mathcal{D} = \mathcal{D}_d(n, \Gamma, \Pi, Enc)$ on $n(d)$-row databases is $(\alpha, T_{\mathrm{San}}(d), \mathcal{C}_{\Gamma}^{(d)})$-hard-to-sanitize as symmetric relaxed synthetic data.

By Theorem 3.7, the family of k-clause CNF formulas, for some constant k, is $(1/2 + \alpha)$-hard-to-approximate under Levin reductions for $\alpha > 0$.

Proof. Let Γ be the assumed family of nice $q(d) -$ CSPs that is $(\alpha + 1/2)$-hard-to-approximate under (possibly inefficient) Levin reductions with length blowup L. Let R, Enc, Dec be the functions corresponding to the Levin reduction to Γ and $\gamma = \gamma(d) \in (\alpha, 1]$ be the parameter from Definition 3.4.

We will use the same approach as in the proof of Theorem 4.4, which is to show that the underlying synthetic database cannot contain a row that satisfies too many clauses of $\varphi_{C_{vk}}$, in order to show that g must map a small input to a large output and a large input to a small answer, contradicting the monotonicity of g.

Let $\varphi_{C_{vk}} = \{\varphi_1, \ldots, \varphi_m\}$, then $\frac{1}{m}\sum_{j=1}^{m} \varphi_j(D) \geq \gamma$. It must also be that $\frac{1}{m}\sum_{j=1}^{m} \varphi_j(\widehat{D}) \leq \gamma - \alpha - 1/2$. Otherwise, there would exist a row $\hat{x} \in \widehat{D} = \mathcal{A}(D)$ such that $\mathrm{val}(\varphi_{C_{vk}}) \geq \gamma - \alpha - 1/2$. But if this were the case we could PCP-decode $\hat{x}$ as in the proof of Theorem 4.4. Thus,

$$\frac{1}{m}\sum_{j=1}^{m} \left(\varphi_j(D) - \varphi_j(\widehat{D})\right) \geq \alpha + 1/2$$

so there must exist $J \in [m]$ s.t. $\varphi_J(D) - \varphi_J(\widehat{D}) \geq \alpha + 1/2$. Since $\varphi_J(\widehat{D}) \geq 0$, we also have $\varphi_J(D) \geq \alpha + 1/2 > 1/2$, and since $\varphi_J(D) \leq 1$ we also have $\varphi_J(\widehat{D}) \leq 1/2 - \alpha < 1/2$.

By monotonicity of g and α-accuracy of $\widehat{D}$ as symmetric relaxed synthetic data, we have

$$g(1/2 - \alpha) \geq g(\varphi_J(\widehat{D})) \geq \varphi_J(D) - \alpha \geq \frac{1}{2}.$$

Consider the negation of φ_J. Since $\neg\varphi_J(D) = 1 - \varphi_J(D)$, we can conclude that $\neg\varphi_J(D) \leq 1/2 - \alpha$ and $\neg\varphi_J(\widehat{D}) \geq 1/2 + \alpha$. Thus, we have

$$g(1/2 + \alpha) \leq g(\neg\varphi_J(\widehat{D})) \leq \neg\varphi_J(D) + \alpha \leq \frac{1}{2}.$$

But $g(1/2 - \alpha) \geq 1/2 \geq g(1/2 + \alpha)$ and $\alpha > 0$ contradicts the monotonicity of g. □

5.3. *Hardness of Sanitizing Very Hard CSPs with Relaxed Synthetic Data*

In this section, we show that no efficient sanitizer can produce accurate relaxed synthetic data for a sequence of CSPs that is $(1 - \mathrm{negl}(d))$-hard-to-approximate under inefficient Levin reductions. By Theorem 3.7, these CSPs include 3-CNF formulas of $\omega(\log d)$ clauses.

Intuitively, an efficient sanitizer must produce a synthetic database of $\hat{n}(d) = \mathrm{poly}(d)$ rows, and thus as d grows, an efficient sanitizer cannot produce a synthetic database that contains a row satisfying a non-negligible fraction of clauses from a particular CSP instance (the signature verification CSP from our earlier results). Thus, using evaluators of the type in Definition 5.1 there can only be one answer to most queries, and thus we cannot get an accurate sanitizer.

Theorem 5.7. *Let $\Gamma = (\Gamma_d)_{d\in\mathbb{N}}$ be a family of nice (Definition 3.2) $q(d)$-CSPs such that $\Gamma_d \cup \neg\Gamma_d$ is $(1 - \varepsilon, 1)$-hard-to-approximate under (possibly inefficient) Levin reductions with length blowup L for a negligible function $\varepsilon = \varepsilon(d)$, and let Enc be the promised encoder. Assume the existence of a $(T_{\mathrm{For}}, Q_{\mathrm{For}})$-digital signature scheme Π*

with verification key length L_{vk} and verification time T_{Ver}. Let $\kappa = \kappa(d)$ be the largest integer such that $L(T_{\mathrm{Ver}}(\kappa)) \leq d/2$ and $L_{\mathrm{vk}}(\kappa)q(d) \leq d/2$ as in the construction of $\mathcal{D}$. Let $T_{\mathrm{San}}, n : \mathbb{N} \to \mathbb{N}$ be any functions such that for every $a > 0$

$$n(d) \cdot d^a + T_{\mathrm{San}}(d) = o(T_{\mathrm{For}}(\kappa(d)))$$

and for every $d \in \mathbb{N}$

$$n(d) \leq Q_{\mathrm{For}}(\kappa(d)).$$

Then, the distribution $\mathcal{D} = \mathcal{D}_d(n, \Gamma, \Pi, Enc)$ on $n(d)$-row databases is $(1/3, T_{\mathrm{San}}(d), \mathcal{C}_\Gamma^{(d)})$-hard-to-sanitize as relaxed synthetic data.

Proof. Let Γ be the assumed family of nice $q(d) - \mathrm{CSPs}$ that is $(1 - \varepsilon(d), 1)$-hard-to-approximate under (possibly inefficient) Levin reductions with length blowup L. Let R, Enc, Dec be the functions corresponding to the Levin reduction to Γ. Let $D \leftarrow_{\mathrm{R}} \mathcal{D}_d$, and $\mathcal{A}(D) = \widehat{D}$.

Let $\varphi_{C_{vk}} = \{\varphi_1, \ldots, \varphi_m\}$. By the construction of $\mathcal{D}_d$, we have

$$\varphi_j(D) = 1$$

for every $j \in [m]$. As in the proof of Theorem 5.6, it must be that

$$\frac{1}{m}\sum_{j=1}^{m} \varphi_j(\widehat{D}) \leq \varepsilon(d).$$

Otherwise, there would exist a row $\hat{x} \in \widehat{D} = \mathcal{A}(D)$ such that $\mathrm{val}(\varphi_{C_{vk}}) \geq \varepsilon(d)$. But if this were the case we could PCP-decode $\hat{x}$ as in the proof of Theorem 4.4.

Since $\mathbb{E}_{j \leftarrow_{\mathrm{R}}[m]}[\varphi_j(\widehat{D})] \leq \varepsilon(d)$, there must exist a subset $J \subseteq [m]$ of size $|J| \geq 2m/3$ such that for all $j \in J$, $\varphi_j(\widehat{D}) \leq 3\varepsilon(d) \leq \mathrm{negl}(d)$.

Since $\widehat{D} \in (\{0,1\}^d)^{\hat{n}}$ for $\hat{n} = \hat{n}(d) = \mathrm{poly}(d)$ (by the efficiency of $\mathcal{A}$),

$$\frac{1}{\hat{n}}\sum_{i=1}^{\hat{n}} \varphi_j(\hat{x}_i) = \varphi_j(\widehat{D}) \in \left\{0, 1/\hat{n}(d), 2/\hat{n}(d), \ldots, 1\right\}$$

and thus $\varphi_j(\widehat{D}) \leq \mathrm{negl}(d)$ implies $\varphi_j(\widehat{D}) = 0$ for large n. Let $\mathcal{E}(\widehat{D}, c) = f(c(\hat{x}_1), \ldots, c(\hat{x}_{\hat{n}}), c)$. If we assume $\widehat{D}$ is $1/3$-accurate as relaxed synthetic data then $f(0^{\hat{n}}, \varphi_j) \geq 2/3$ for every $j \in J$.

Now consider the execution of $\mathcal{A}$ on the database $D' = (0^d)^n$. With probability $1 - \mathrm{negl}(d)$, $Dec(0^d, C_{vk})$ is not a valid message–signature pair, thus by Definition 3.4 Part 3, we have

$$\varphi_{C_{vk}}(D') = \varphi_{C_{vk}}(0^d) \leq \varepsilon(d).$$

Since the rows of D' are identical, $\varphi_j(D') \in \{0, 1\}$ for every $j \in [m]$. So for at least a $1 - \varepsilon(d)$ fraction of $j \in [m]$, we have $\varphi_j(D') = 0$.

Let $\widehat{D}' = \mathcal{A}(D')$. By repeating the signature-forging argument, we see that with probability $1 - \mathrm{negl}(d)$

$$\frac{1}{m}\sum_{j=1}^{m}\varphi_j(\widehat{D}') \leq \varepsilon(d)$$

and thus there must exist a subset $J' \subseteq [m]$ of size $|J'| \geq 2m/3$ such that $j \in J' \Longrightarrow \varphi_j(\widehat{D}') \leq 3\varepsilon(d) \leq \mathrm{negl}(d)$. So $\varphi_j(\widehat{D}') = 0$ for every $j \in J'$ as well. There must also exist a set $J'' \subseteq J'$ of size $|J''| = (2/3 - \varepsilon(d))m$, such that for every $j \in J''$, $\varphi_j(D') = \varphi_j(\widehat{D}') = 0$. So if $\widehat{D}'$ is 1/3-accurate for $\mathcal{C}$ as relaxed synthetic data it must be that $f(0^{\hat{n}}, \varphi_j) \leq 1/3$ for every $j \in J''$.

By our choice of J and J'', there must exist $j \in J \cap J''$ such that:

1. $f(0^{\hat{n}}, \varphi_j) \geq 2/3$, and
2. $f(0^{\hat{n}}, \varphi_j) \leq 1/3$,

which is a contradiction. □

5.4. *Positive Results for Relaxed Synthetic Data*

Our hardness results for relaxed synthetic data suggest that releasing private relaxed synthetic data may be no easier than releasing synthetic data. In this section, we show that this is not the case, by presenting an efficient, accurate sanitizer for families of constant-arity parities that outputs symmetric relaxed synthetic data and show that this sanitizer also yields accurate answers for any family of constant-arity predicates when evaluated as a relaxed synthetic data. Our result for parities shows that relaxed synthetic data (and even symmetric relaxed synthetic data) allows for more efficient sanitization than standard synthetic data, since Theorem 4.4 rules out an accurate, efficient sanitizer that produces a standard synthetic database, even for the class of 2-literal parity predicates. Our result for parities also shows that our hardness result for symmetric relaxed synthetic data (Theorem 5.6) is tight with respect to the required hardness of approximation, since the class of 3-literal parity predicates is $(1/2 - \varepsilon)$-hard-to-approximate [23]

A function $f : \{0, 1\}^d \to \{0, 1\}$ is a *k-junta* if it depends on at most k variables. Let $\mathcal{J}_{d,k}$ be the set of all k-juntas on d variables.

Theorem 5.8. *There exists an ε-differentially private sanitizer that runs in time* $\mathrm{poly}(n, d, |\mathcal{J}_{d,k}|)$ *and produces relaxed synthetic data and is (α, β)-accurate for $\mathcal{J}_{d,k}$ when*

$$n \geq \frac{C\binom{d}{\leq k}\log\left(\binom{d}{\leq k}/\beta\right)}{\alpha\varepsilon}$$

for a sufficiently large constant C, where $\binom{d}{\leq k} = \sum_{i=0}^{k}\binom{d}{i}$.

The privacy, accuracy, and efficiency guarantees of our theorem can be achieved without relaxed synthetic data simply by releasing a vector of noisy answers to the queries

[13]. Our sanitizer will, in fact, begin with this vector of noisy answers and construct relaxed synthetic data from those answers. Our technique is similar to that of Barak et al. [3], which begins with a vector of noisy answers to *parity queries* (defined in Sect. 5.4.1) and constructs a (standard) synthetic database that gives answers to each query that are close to the initial noisy answers. They construct their synthetic database by solving a linear program over 2^d variables that correspond to the frequency of each possible row $x \in \{0,1\}^d$, and thus their sanitizer runs in time exponential in d. Our sanitizer also starts with a vector of noisy answers to parity queries and *efficiently* constructs symmetric relaxed synthetic data that gives answers to each query that are close to the initial noisy answers after applying a *fixed linear scaling*. We then show that the database our sanitizer constructs is also accurate for the family of k-juntas using an affine shift that depends on the junta.

5.4.1. *Efficient Sanitizer for Parities*

To prove Theorem 5.8, we start with a sanitizer for *parity predicates*.

Definition 5.9. *(Parity Predicate)* A function $\chi : \{0,1\}^d \to \{-1,1\}$ is a *parity predicate*[6] if there exists a vector $s \in \{0,1\}^d$ s.t.

$$\chi(x) = \chi_s(x) = (-1)^{\langle x,s \rangle}.$$

We will use $wt(s) = \sum_{i=1}^d s_i$ to denote the number of nonzero entries in s.

Theorem 5.10. *Let $\mathcal{P}$ be a family of parity predicates on d variables such that $\chi_{0^d} \notin \mathcal{P}$. There exists an ε-differentially private sanitizer $\mathcal{A}(D,\mathcal{P})$ that runs in time* $\mathrm{poly}(n,d)$ *and produces symmetric relaxed synthetic data that is (α,β)-accurate for $\mathcal{P}$ when*

$$n \geq \frac{2|\mathcal{P}|\log(2|\mathcal{P}|/\beta)}{\alpha\varepsilon}.$$

The analysis of our sanitizer will make use of the following standard fact about parity predicates

Fact 5.11. *Two parity predicates $\chi_s, \chi_t : \{0,1\}^d \to \{-1,1\}$ are either identical or orthogonal. Specifically, for $s \neq t$, $s \neq 0^d$ and $b \in \{-1,1\}$,*

$$\mathop{\mathbb{E}}_{x \leftarrow_R \{0,1\}^d}[\chi_s(x) | \chi_t(x) = b] = \mathop{\mathbb{E}}_{x \leftarrow_R \{0,1\}^d}[\chi_s(x)] = 0.$$

Our sanitizer will start with noisy answers to the predicate queries $\chi_s(D)$. Each noisy answer will be the true answer perturbed with noise from a *Laplace distribution*. The Laplace distribution $Lap(\sigma)$ is a continuous distribution on $\mathbb{R}$ with probability density

[6] In the preliminaries, we define a predicate to be a $\{0,1\}$-valued function, but our definition naturally generalizes to $\{-1,1\}$-valued functions. For $c : \{0,1\}^d \to \{-1,1\}$ and database $D = (x_1,\ldots,x_n) \in (\{0,1\}^d)^n$, we define $c(D) = \frac{1}{n}\sum_{i=1}^n c(x_i)$.

function $p_\sigma(y) \propto \exp(-|y|/\sigma)$. The following theorem of Dwork et al. [13] shows that these queries are differentially private for an appropriate choice of σ.

Theorem 5.12. ([13]) *Let $(c_1, c_2, \ldots, c_k)$ be a set of predicates and let $\sigma = k/n\varepsilon$ and let $D \in (\{0,1\}^d)^n$ be a database. Then, the mechanism $\mathcal{A}(D) = (c_1(D) + Z_1, c_2(D) + Z_2, \ldots, c_k(D) + Z_k)$, where $(Z_1, \ldots, Z_k)$ are independent samples from $Lap(\sigma)$, is ε-differentially private.*

In order to argue about the accuracy of our mechanism, we need to know how much error is introduced by noise from the Laplace distribution. The following fact gives a bound on the tail of a Laplace random variable.

Fact 5.13. *The tail of the Laplace distribution decays exponentially. Specifically,*

$$\Pr[|Lap(\sigma)| \geq t] = \exp(-t/\sigma).$$

Now, we present our sanitizer for queries that are parity functions. We will not consider the query χ_{0^d} as $\chi_{0^d}(x) = 1$ for every $x \in \{0,1\}^d$. Let $\mathcal{P}$ be a set of parity functions that does not contain χ_{0^d}. We now present a poly$(n, d, |\mathcal{P}|)$-time sanitizer for $\mathcal{P}$.

Our sanitizer starts by getting noisy estimates of the quantities $\chi(D)$ for each predicate $\chi \in \mathcal{P}$ by adding Laplace noise. Then, it builds the relaxed synthetic data $\widehat{D}$ in blocks of rows. Each block of rows is "assigned" to contain an answer to a query χ. In that block, we randomly choose rows such that the expected value of χ on each row equals the noisy estimate of $\chi(D)$. By Fact 5.11, the expected value of every other predicate χ' is 0 for rows in this block. The sanitizer is accurate so long as the total number of rows is sufficient for the value of $\chi(\widehat{D})$ to be concentrated around its expectation.

```
Sanitizer $\mathcal{A}(D, \mathcal{P})$, where $P = \{\chi^{(1)}, \ldots, \chi^{(t)}\}$:
  Let $\sigma := |\mathcal{P}|/n\varepsilon \quad T := (2|\mathcal{P}|/\alpha^2)\log(4|\mathcal{P}|/\beta)$
  for all $j = 1, \ldots, t$ do
    Let $a_j := \chi^{(j)}(D) + Lap(\sigma)$
    for $i = jT + 1$ to $(j+1)T$ do
      With probability $(a_j + 1)/2$: Let $\hat{x}_i \leftarrow_{\mathrm{R}} \{x \in \{0,1\}^d \mid \chi^{(j)}(x) = 1\}$
      Otherwise: Let $\hat{x}_i \leftarrow_{\mathrm{R}} \{x \in \{0,1\}^d \mid \chi^{(j)}(x) = -1\}$
    end for
  end for
  return $\widehat{D} = (\hat{x}_1, \ldots, \hat{x}_{tT})$

Evaluator $\mathcal{E}_{\mathcal{P}}(\widehat{D}, \chi)$:
  return $|\mathcal{P}| \cdot \chi(\widehat{D})$
```

The following claims will suffice to establish Theorem 5.10

Claim 5.14. *$\mathcal{A}$ is ε-differentially private.*

Proof. The output of $\mathcal{A}$ only depends on the answer to $|\mathcal{P}|$ predicate queries. By Theorem 5.12, the answers to $|\mathcal{P}|$ predicate queries perturbed by independent samples from $Lap(|\mathcal{P}|/n\varepsilon)$ is ε-differentially private. □

Claim 5.15. *$\mathcal{A}$ is (α, β)-accurate for $\mathcal{P}$ when*

$$n \geq \frac{2|\mathcal{P}| \log(2|\mathcal{P}|/\beta)}{\alpha\varepsilon}.$$

Proof. We want to show that for every $\chi^{(j)} \in \mathcal{P}$

$$\left| |\mathcal{P}| \cdot \chi^{(j)}(\widehat{D}) - \chi^{(j)}(D) \right| \leq \alpha$$

except with probability β. To do so, we consider separately the error introduced in going from $\chi^{(j)}(D)$ to a_j using Laplacian noise and the error introduced in going from noisy answers a_j to $\chi^{(j)}(\widehat{D})$ by sampling rows at random.

First, we bound the error introduced by the noisy queries to D. Specifically, we want to show that for every $\chi^{(j)} \in \mathcal{P}$

$$\left| \chi^{(j)}(D) - a_j \right| \leq \alpha/2$$

except with probability $\beta/2$. For each $\chi^{(j)}$, we have

$$\Pr[|\chi^{(j)}(D) - a_j| \geq \alpha/2] \leq \exp(-n\varepsilon\alpha/2|\mathcal{P}|)$$

by Fact 5.13. So by a union bound, we have

$$\Pr[\exists\chi^{(j)}\ |\chi^{(j)}(D) - a_j| \geq \alpha/2] \leq |\mathcal{P}| \exp(-\alpha/2\sigma) \leq |\mathcal{P}| \exp(-n\varepsilon\alpha/2|\mathcal{P}|) < \beta/2,$$

so long as

$$n \geq \frac{2|\mathcal{P}| \log(2|\mathcal{P}|/\beta)}{\alpha\varepsilon}.$$

We also want to show that for every $\chi^{(j)} \in \mathcal{P}$

$$\left| |\mathcal{P}| \cdot \chi^{(j)}(\widehat{D}) - a_j \right| \leq \alpha/2$$

except with probability $\beta/2$, where a_j is the noisy answer for $\chi^{(j)}(D)$ computed in $\mathcal{A}(D)$. To do so, we will show that the expectation of $|\mathcal{P}|\chi^{(j)}(\widehat{D})$ is indeed a_j, then we will use a Chernoff–Hoeffding bound to show that the random rows generated by $\mathcal{A}(D)$ are close to their expectation. Finally we take a union bound over all $\chi \in \mathcal{P}$.

Fix $\chi^{(j)} \in \mathcal{P}$ and consider $\chi^{(j)}(\widehat{D})$. $\chi^{(j)}(\widehat{D})$ is the sum of T independent biased coin flips. In rows $jT+1, jT+2, \ldots, (j+1)T$ (the rows where we focus on $\chi^{(j)}$) the expectation of each coin flip is a_j, and in all other rows the expectation of each coin flip is 0 by Fact 5.11. Thus,

$$\mathbb{E}\left[\chi^{(j)}(\widehat{D})\right] = \mathbb{E}\left[\frac{1}{\hat{n}}\sum_{i=1}^{\hat{n}} \chi^{(j)}(\hat{x}_i)\right] = a_j/|\mathcal{P}|$$

for every $\chi^{(j)} \in \mathcal{P}$.

By a Chernoff–Hoeffding bound,[7] we conclude

$$\Pr\left[\left||\mathcal{P}| \cdot \chi^{(j)}(\widehat{D}) - a_j\right| \geq \alpha/2\right] < 2\exp(-T\alpha^2/2|\mathcal{P}|).$$

By taking a union bound over $\mathcal{P}$, we conclude

$$\Pr\left[\exists\chi^{(j)}\ \left||\mathcal{P} \cdot |\chi^{(j)}(\widehat{D}) - a_j\right| \geq \alpha/2\right] < 2|\mathcal{P}|\exp(-T\alpha^2/2|\mathcal{P}|) \leq \beta/2.$$

Combining the two bounds suffices to prove the claim. □

5.4.2. *Efficient Sanitizer for k-Juntas*

We now show that our sanitizer for parity queries can also be used to give accurate answers for any family of k-juntas, for constant k. We start with the observation that k-juntas only have Fourier mass on coefficients of weight at most k. Alternatively, this says that any k-junta can be written as a linear combination of parity functions on at most k variables. (In the language of our previous construction, χ_s such that $wt(s) \leq k$.) Thus, we start by running our sanitizer for parity predicates on the set $\mathcal{P}_k$ containing all parity predicates on at most k variables. We have to modify the evaluator function to take into account that not every k-junta predicate has the same bias. Indeed, we cannot control $\chi_{0^d}(\widehat{D})$ in our output, as $\chi_{0^d}(D) = 1$ for any database. Thus, our evaluator will apply an affine shift to each result that depends on the junta. Because the evaluator depends on the predicate, the resulting sanitizer no longer outputs symmetric relaxed synthetic data.

The use of a sanitizer for parity queries as a building block to construct a sanitizer for arbitrary k-juntas is inspired by [3], which uses a noisy vector of answers to parity queries as a building block to construct synthetic data for a particular class of k-juntas (conjunctions on k-literals). However, while their sanitizer constructs a standard synthetic database and is inefficient, our construction of symmetric relaxed synthetic data for parity predicates is efficient, and thus our eventual sanitizer for k-juntas will also be efficient.

[7] One form of the Chernoff–Hoeffding bound states if $X_1, \ldots, X_n$ are independent random variables over $[0, 1]$ and $X = (1/n)\sum_{i=1}^{n}$ then $\Pr[|X - \mathbb{E}[X]| \geq t] < 2\exp(-2nt^2)$ [12].

Consider a predicate $c : \{0,1\}^d \to \{0,1\}$. Then, we can take the Fourier expansion

$$c(x) = \sum_{s \in \{0,1\}^d} \widehat{c}(s) \chi_s(x)$$

where

$$\widehat{c}(s) = \mathop{\mathbb{E}}_{x \leftarrow_{\mathrm{R}} \{0,1\}^d} [c(x)\chi_s(x)].$$

The accuracy of our sanitizer relies on the following fact about the Fourier coefficients of k-juntas

Fact 5.16. *If $c : \{0,1\}^d \to \{0,1\}$ is a k-junta, then $wt(s) > k \Longrightarrow \widehat{c}(s) = 0$*

Let $\mathcal{P}_k = \left\{\chi_s \mid s \in \{0,1\}^d, 1 \le wt(s) \le k\right\}$. Our sanitizer for k-juntas is just the sanitizer for parities applied to the set $\mathcal{P}_k$. We now define the evaluator that computes the answer to conjunction queries from the output of $\mathcal{A}(D, \mathcal{P}_k)$.

Evaluator $\mathcal{E}(\widehat{D}, c)$ for a k-junta c:

return $|\mathcal{P}_k| c(\widehat{D}) - (|\mathcal{P}_k| - 1)\widehat{c}(\emptyset)$

Efficiency and privacy follow from the analysis of $\mathcal{A}$. Let $\mathcal{J}_{d,k}$ be a family of all k-juntas on d variables.

Theorem 5.17. *$\mathcal{A}(D, \mathcal{P}_k)$ is (α, β)-accurate for $\mathcal{J}_{d,k}$ using $\mathcal{E}$ when*

$$n \ge \frac{2|\mathcal{P}_k| \log(2|\mathcal{P}_k|/\beta)}{\alpha\varepsilon}.$$

Proof. Let $c \in \mathcal{J}_{d,k}$ be a fixed predicate. Assume that $\widehat{D}$ is α-accurate for $\mathcal{P}_k$ using $\mathcal{E}_{\mathcal{P}}$. This event occurs with probability at least $1 - \beta$ by Theorem 5.10 and our assumption on n. We now analyze the quantity $c(\widehat{D})$.

$$\begin{aligned} c(\widehat{D}) &= \frac{1}{\hat{n}} \sum_{i=1}^{\hat{n}} c(\hat{x}_i) \\ &= \frac{1}{\hat{n}} \sum_{i=1}^{\hat{n}} \sum_{s \in \{0,1\}^d} \widehat{c}(s) \chi_s(\hat{x}_i) \\ &= \sum_{s \in \{0,1\}^d : |s| \le k} \widehat{c}(s) \chi_s(\widehat{D}) \qquad (3) \\ &= \widehat{c}(\emptyset) + \sum_{s \in \{0,1\}^d : 1 \le |s| \le k} \widehat{c}(s) \chi_s(\widehat{D}) \qquad (4) \end{aligned}$$

$$\leq \widehat{c}(\emptyset) + \sum_{s \in \{0,1\}^d : 1 \leq |s| \leq k} \widehat{c}(s) \left(\frac{\chi_s(D) + \alpha}{|\mathcal{P}_k|} \right)$$

$$\leq \frac{1}{|\mathcal{P}_k|} \left((|\mathcal{P}_k| - 1)\, \widehat{c}(\emptyset) + \sum_{s \in \{0,1\}^d : |s| \leq k} (\widehat{c}(s) \chi_s(D) + \alpha) \right)$$

$$= \frac{1}{|\mathcal{P}_k|} \left((|\mathcal{P}_k| - 1)\, \widehat{c}(\emptyset) + c(D) \right) + \alpha \tag{5}$$

where step 3 uses Fact 5.16, step 4 uses the fact that $\chi_{0^d}(x) = 1$ everywhere, and step 5 uses the fact that $\widehat{D}$ is α-accurate for $\mathcal{P}_k$ when evaluated by $\mathcal{E}_{\mathcal{P}}$. A similar argument shows that

$$c(\widehat{D}) \geq \frac{1}{|\mathcal{P}_k|} \left((|\mathcal{P}_k| - 1)\, \widehat{c}(\emptyset) + c(D) \right) - \alpha$$

Thus, $\widehat{D} = \mathcal{A}(D)$ is α-accurate for $\mathcal{J}_{d,k}$ using $\mathcal{E}$ with probability at least $1 - \beta$. □

Acknowledgements

We thank Boaz Barak, Irit Dinur, Cynthia Dwork, Vitaly Feldman, Oded Goldreich, Johan Håstad, Valentine Kabanets, Dana Moshkovitz, Anup Rao, Guy Rothblum, and Les Valiant for helpful conversations. We are also grateful to the anonymous referees for helpful comments on the presentation of this work.

References

[1] M. Alekhnovich, M. Braverman, V. Feldman, A. R. Klivans, T. Pitassi, The complexity of properly learning simple concept classes. in *J. Comput. Syst. Sci.*, **74**, 16–34, (2008)

[2] L. Babai, L. Fortnow, L.A. Levin, M. Szegedy, Checking computations in polylogarithmic time. in *STOC*, 21–31, (1991)

[3] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, K. Talwar, Privacy, accuracy, and consistency too: A holistic solution to contingency table release. in *Proceedings of the 26th Symposium on Principles of Database Systems*, pp. 273–282, (2007)

[4] B. Barak, O. Goldreich, Universal arguments and their applications. in *SIAM J. Comput.*, vol 38, 1661–1694, (2008)

[5] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, S.P. Vadhan, Robust pcps of proximity, shorter pcps, and applications to coding. in *SIAM J. Comput.*, vol. 36, pages 889–974, (2006)

[6] A. Blum, C. Dwork, F. McSherry, K. Nissim, Practical privacy: The SuLQ framework. in *Proceedings of the 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, (June 2005)

[7] A. Blum, K. Ligett, A. Roth, A learning theory approach to non-interactive database privacy. in *Proceedings of the 40th ACM SIGACT Symposium on Thoery of Computing*, (2008)

[8] K. Chandrasekaran, J. Thaler, J. Ullman, A. Wan, Faster private release of marginals on small databases. in *ITCS* (ACM, New York, 2014), pp. 387–402

[9] N. Creignou, A dichotomy theorem for maximum generalized satisfiability problems. In *J. Comput. Syst. Sci.*, volume 51, pages 511–522, (1995)

[10] I. Dinur, K. Nissim, Revealing information while preserving privacy. in *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 202–210, (2003)

[11] I. Dinur, O. Reingold, Assignment testers: Towards a combinatorial proof of the pcp theorem. in *SIAM J. Comput.*, volume 36, pages 975–1024, (2006)

[12] D.P. Dubhashi, S. Sen, Concentration of measure for randomized algorithms: techniques and applications. in Handbook of Randomized Algorithms, (2001)

[13] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Theory of Cryptography Conference*, pp. 265–284, (2006)

[14] C. Dwork, M. Naor, O. Reingold, G. Rothblum, S. Vadhan, When and how can privacy-preserving data release be done efficiently? In *Proceedings of the 2009 International ACM Symposium on Theory of Computing (STOC)*, (2009)

[15] C. Dwork, A. Nikolov, K. Talwar, Using convex relaxations for efficiently and privately releasing marginals. in *Proceedings of the thirtieth annual symposium on Computational geometry* (ACM, New York, 2014), pp. 261

[16] C. Dwork, K. Nissim, Privacy-preserving datamining on vertically partitioned databases. In *Proceedings of CRYPTO 2004*, vol. 3152, pp. 528–544, (2004)

[17] C. Dwork, A. Roth, The algorithmic foundations of differential privacy, (2014)

[18] C. Dwork, G. Rothblum, S.P. Vadhan, Boosting and differential privacy. in *Proceedings of FOCS 2010*, (2010)

[19] V. Feldman, Hardness of proper learning. in *The Encyclopedia of Algorithms* (Springer, New York, 2008)

[20] V. Feldman, Hardness of approximate two-level logic minimization and PAC learning with membership queries. *Journal of Computer and System Sciences*, 75(1):13–26, (2009)

[21] O. Goldreich, *Foundations of Cryptography*, volume 2. Cambridge University Press, (2004)

[22] M. Hardt, G.N. Rothblum, R.A. Servedio, Private data release via learning thresholds. in *SODA* (SIAM, New York, 2012), pp. 168–187

[23] J. Håstad, Some optimal inapproximability results. in *J. ACM*, volume 48, pages 798–859, (2001)

[24] J. Justesen, On the complexity of decoding reed-solomon codes (corresp). IEEE Trans. Inf. Theory 22(2):237–238 (1976)

[25] M.J. Kearns, L.G. Valiant, Cryptographic limitations on learning boolean formulae and finite automata. in *J. ACM*, volume 41, pages 67–95, (1994)

[26] S. Khanna, M. Sudan, L. Trevisan, D.P. Williamson, The approximability of constraint satisfaction problems. in *SIAM J. Comput.*, vol. 30, pp. 1863–1920, (2000)

[27] J. Kilian, A note on efficient zero-knowledge proofs and arguments (extended abstract). in *STOC*, (1992)

[28] V. Lyubashevsky, D. Micciancio, Asymptotically efficient lattice-based digital signatures. In R. Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science* (Springer, Berlin, 2008), pp. 37–54

[29] S. Micali, Computationally sound proofs. in *SIAM J. Comput.*, volume 30, pages 1253–1298, (2000)

[30] M. Naor, M. Yung, Universal one-way hash functions and their cryptographic applications. in *STOC*, pp. 33–43, (1989)

[31] C.H. Papadimitriou, M. Yannakakis, Optimization, approximation, and complexity classes. in *J. Comput. Syst. Sci.*, volume 43, pages 425–440, (1991)

[32] L. Pitt, L.G. Valiant, Computational limitations on learning from examples. in *J. ACM*, volume 35, pages 965–984, (1988)

[33] J.P. Reiter, J. Drechsler, Releasing multiply-imputed synthetic data generated in two stages to protect confidentiality. Iab discussion paper, Intitut für Arbeitsmarkt und Berufsforschung (IAB), Nürnberg (Institute for Employment Research, Nuremberg, Germany), (2007)

[34] J. Rompel, One-way functions are necessary and sufficient for secure signatures. in *STOC*, pp. 387–394, (1990)

[35] A. Roth, T. Roughgarden, Interactive privacy via the median mechanism. in *STOC 2010*, (2010)

[36] D.A. Spielman, Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, **42**(6):1723–1731, (1996)

[37] J. Thaler, J. Ullman, S.P. Vadhan, Faster algorithms for privately releasing marginals. in *ICALP (1)* (Springer, Berlin, 2012), pp. 810–821

[38] L.G. Valiant, A theory of the learnable. *Communications of the ACM*, **27**(11):1134–1142, (1984)