Certifying Private Probabilistic Mechanisms

Zoë Ruha Bell
UC Berkeley

Shafi Goldwasser
UC Berkeley

Michael P. Kim
Cornell University[*]

Jean-Luc Watson
UC Berkeley[†]

June 11, 2024[‡]

Abstract

In past years, entire research communities have arisen to address concerns of privacy and fairness in data analysis. At present, however, the public must trust that institutions will re-implement algorithms voluntarily to account for these social concerns. Due to additional cost, widespread adoption is unlikely without effective legal enforcement. A technical challenge for enforcement is that the methods proposed are often *probabilistic mechanisms*, whose output must be drawn according to precise, and sometimes secret, distributions. The Differential Privacy (DP) case is illustrative: if a cheating curator answers queries according to an overly-accurate mechanism, privacy violations could go undetected. The need for effective enforcement raises the central question of our paper: *Can we efficiently certify the output of a probabilistic mechanism enacted by an untrusted party?* To this end:

1. We introduce two new notions: *Certified Probabilistic Mechanisms* (CPM) and *Random Variable Commitment Schemes* (RVCS). A CPM is an interactive protocol that forces a prover to enact a given probabilistic mechanism or be caught; importantly, the interaction does not reveal secret parameters of the mechanism. An RVCS—a key primitive for constructing CPMs—is a commitment scheme where the verifier is convinced that the commitment is to an RV sampled according to an agreed-upon distribution, but learns nothing else.
2. We instantiate the general notion of CPM for the special case of Certifying DP. We build a lightweight, doubly-efficent interactive proof system to certify arbitrary-predicate counting queries released via the DP Binomial mechanism. The construction relies on a commitment scheme with perfect hiding and additive homomorphic properties that can be used to release a broad class of queries about a committed database, which we construct on top of Pedersen commitments.
3. Finally, we demonstrate the immediate feasibility of Certified DP via a highly-efficient and scalable prototype implementation to answer counting queries of arbitrary predicates. The mechanism is composed of an offline and online stage, where the online phase allows for non-interactive certification of queries. For example, we show that CDP queries over a US Census Public Use Microdata Sample (PUMS) [23] ($n = 7000$) can be completed in only 1.6 ms and verified in just *38 μs.* Our implementation is available in open source at `https://github.com/jlwatson/certified-dp`.

[*]Much of this work completed at the Miller Institute for Basic Research in Science and the Simons Institute for the Theory of Computing at UC Berkeley.

[†]Completed at Sky Computing Lab, which is supported by gifts from Accenture, AMD, Anyscale, Google, IBM, Intel, Microsoft, Mohamed Bin Zayed University of Artificial Intelligence, Samsung SDS, SAP, Uber, and VMware

[‡]© IACR 2024. This article is a minor revision of the version published by Springer-Verlag.

Contents

1 Introduction

"Trust" in data-driven systems is emerging as a key concern within computer science research and across society as a whole. While recent years have seen amazing progress of the functionality of data-driven systems, they also provide ample reasons to be wary. As a pertinent example, general-purpose generative language models [34, 68, 20, 70, 21] are demonstrating remarkable capabilities across diverse tasks, but also raise serious concerns. These models infamously regurgitate misinformation and abusive content, and present threats to indivduals' privacy and copyrighted material [13, 83]. Whether we decide to use such models often boils down to the question: *Can we trust the sources, design, and output of the model?*

At the same time, new research communities—studying topics like privacy [38], fairness [35, 53, 54], and robustness [69, 31, 57]—have popped up to address the growing need for "trustworthy" systems for data science and machine learning.[1] The typical research program identifies flaws in the existing systems, and then proposes a re-designed system using new algorithms that ensure (or at least steer towards) "good" outputs, avoiding the "bad." When employed correctly, these novel algorithms can provide strong, formal guarantees along critical dimensions (much in the way that cryptography provides guarantees of security).

And yet, despite the progress in proposing methods that will increase trustworthiness, it is far from clear that platforms will voluntarily deploy these new methods. In any re-design, the new algorithms incur a cost along various dimensions, including lower perceived accuracy or functionality, heavier use of data, time, and memory, as well as the human efforts required to implement the changes faithfully. While some companies have declared their intentions to employ privacy and fairness practices, for-profit corporations are typically left in charge of determining when, how, and to what degree to use these measures—measures that may go against their bottom line. For instance, Meta has only recently agreed to re-implement part of their ad delivery platform to be more fair, as part of a settlement of a major discrimination lawsuit [2].

Some in the community hope that, going forward, litigation and regulation will play a key role in governing data-driven algorithms. Of course, legal regulations without effective technical solutions to certify compliance will be impossible to enforce. The cryptographic paradigm for certifying compliance is for institutions to *prove* to auditors that they are utilizing mechanisms for trustworthiness at design-time. Indeed, the broad strategy of employing (sometimes zero-knowledge) cryptographic proofs has been investigated in a variety of legal domains [8, 16]. While this methodology is appealing in its generality, it is unlikely to be adopted widely. General-purpose certification tools such as MPC and zero-knowledge proofs for NP simply cost too much. A key challenge for broader adoption is whether we can overcome this prohibitive cost in practical use cases.

Another challenge of a more fundamental nature is that many of the algorithms we'd like to certify are *probabilistic mechanisms.* That is, not only do the algorithms rely on some internal randomness, but their output must be random, drawn according to precise, sometimes private, distributions in order for the claimed properties (of privacy, fairness, etc.) to hold. Consider, for instance, the property of *differential privacy* [38]. Differential Privacy (DP) guarantees individual data privacy in statistical analyses by adding carefully-constructed *private* randomness to the output of statistical

[1]We use the term "trustworthiness" colloquially to include a wide class of properties, including privacy, fairness, and robustness.

queries. A mechanism that proves DP compliance by certifying the internal randomness of the algorithm runs the risk of violating DP in the process. Similar issues of randomness as a first-order property of the output show up in other areas of trustworthy ML, including sampling fair classifications based on predicted probabilities [3, 81, 5] and in designing prediction strategies that are robust to adversarial manipulation [31, 48].

The Case of Differential Privacy. No longer a purely-academic endeavor, differential privacy (DP) is employed voluntarily across major organizations handling individual data, from tech giants like Google [51] and Apple [1] to government agencies like the Census Bureau [22]. Employing differential privacy, however, is not free. Given a fixed set of data, DP mechanisms are inherently noisier and less accurate than their non-private counterparts. In settings where accuracy is deemed paramount by clients, database curators may hold an incentive to deviate from the promised DP mechanism.[2] Rather than answering queries under an appropriate noise distribution, the curator may choose to answer queries according to an overly-accurate mechanism that violates privacy.

In other settings, adding excessive noise to "ensure user privacy" may serve as a means to evade effective scrutiny. For instance, the terms of Meta's discrimination settlement stipulate that they must report statistics about their platforms to third-party auditors [2]. A major outstanding question in this case is how reporting will be handled, so that Meta can earnestly protect their users' privacy and auditors can be convinced of the veracity of the reporting.

In existing frameworks for DP, regulators have little recourse against data curators suspected of bad behavior. Privacy auditing techniques have gained significant attention [76], but typically rely on control of the curator's algorithm. Pushing for guidance and accountability, some leaders in the privacy community have called for an "Epsilon Registry," where plaforms would voluntarily publish their privacy parameters [37, 33]. In the current ecosystem, however, proper accounting of privacy parameters requires painstaking scientific effort [77].

This Work. Across the growing landscape of algorithmic solutions for trustworthy computation, the state of affairs begs for a new cryptographic primitive which is the central topic of this paper: **an efficient tool to certify the output of probabilistic mechanisms**, based on a *single* draw from the mechanism.

1.1 Our contributions

In this work, we initiate the study of *Certified Probabilistic Mechanisms*, which allow a prover to convince a verifier of the validity of a random output from some agreed-upon, but private, probabilistic mechanism. We will focus on constructing special-purpose efficient certified probabilistic mechanisms, aiming at eventual adoption of certification as the method of choice to enforce compliance. We detail our contributions next.

[2] Indeed, much of the pushback against the use of DP by the US Census Bureau came from social scientists who questioned the utility of noisy synthetic data release [74].

Certified Probabilistic Mechanisms. A primary contribution is to articulate and define the semantics of Certified Probabilistic Mechanisms (CPMs); indeed, even defining what it means to certify that a single realization of a probabilistic mechanism looks "right" is a subtle task. We start by considering a (non-certified) probabilistic mechanism M. Given input parameters θ (e.g., proprietary ML model parameters, or a database of individual records), $\mathsf{M}(\theta)$ returns a random output based on the internal randomness of M.

A **Certified Probabilistic Mechanism** (CPM) is an interactive protocol between a prover and verifier satisfying three properties.

- **Correctness**: *An honest verifier, after interacting with an honest prover, outputs a random value drawn according to the intended mechanism* $\mathsf{M}(\theta)$.
- **Cheating-Prover Soundness**: *When the Prover enacts a mechanism that deviates significantly from* $\mathsf{M}(\theta)$*, the honest Verifier detects, returning* $\perp$ *with probability equal to this deviation.*
- **Cheating-Verifier Soundness**: *Even if the Verifier deviates, the output of the protocol is drawn according to* $\mathsf{M}(\theta)$ *(or results in failure), and keeps the Prover's input* θ *and internal randomness hidden.*

Our definition of CPMs is carefully designed to allow the verifier to be convinced of the integrity of the outcome they observe, and learn nothing more. The ability to control what information is and isn't released through the certification procedure is critical for this notion to be used in legally-mandated certification.

Certified DP. With the definition of general CPMs in place, we focus our attention on the practically-relevant special case of certifying a database curator's claimed use of a differentially private mechanism. We work in the setting of a single untrusted data curator, and focus on certifying the proper release of DP statistics, catching the curator when DP violations occur rather than aiming to *prevent* DP violations (which is impossible in the single data curator case).

Prior Work. The question of certifying DP mechanisms has been considered in a few prior works. Most relevant to our work, Biswas and Cormode [15] study "Interactive Proofs for DP" in an MPC setup. First, they focus on a multi-curator model where curator clients secret-share their input data to several different provers (at least one of which is assumed to be honest), who wish to protect the privacy of the data from each other, in addition to guaranteeing DP answers (or detection of a violation) to queries by a client/verifier on the aggregated data. In contrast, our goal is to certify the trustworthiness of a single curator by guaranteeing the detection of DP violations by this curator. Second, we remark that the definition of [15] when applied to the single curator case is impossible to achieve: they require the verifier to detect DP deviations when the curator output is *not perfectly* equal to the DP mechanism output. In contrast, we require the *statistical difference* between the curator output and the the DP mechanism output to be equal to the verifier's detection probability. Essentially, they formally treat the mechanism output as a deterministic function whereas we view it as a sample of a probabilistic process. Finally, an important issue that we deal with (unlike [15]) is that the mere fact that a verifier learns that a prover/curator is cheating and DP violations take place is in itself a DP violation. We take this into account in our analysis (see Remark 3.3 and Theorem 3.5).

Certified DP Queries via Random Variable Commitments. Our main technical contribution is to build a lightweight interactive proof system for certifying the DP release of predicate counting queries. At its core, the CPM we design is simple to state: the curator, a.k.a., the *prover*, commits to their database D in a way that allows them to open counting queries $f(D)$ over the database; additionally, the prover *commits to the noise* $\mathbf{Z}$ drawn according to an appropriate noise distribution; finally, the prover homomorphically adds $f(D)+\mathbf{Z}$ and sends the result (and opening) to the analyst, a.k.a., the *verifier*. This can be cast as a "commit-and-prove" approach [12]. But, wait! What does it mean for the prover to "commit to the noise" that they'll use to ensure DP? Of course, if the prover samples the randomness on their own, it will be impossible to guarantee soundess. On the other hand, having the verifier choose the randomness runs the risk of exposing the randomness and thus, exposing more information about $f(D)$ than intended by the original probabilistic mechanism.

To solve this quandry, we introduce the idea of a ***random variable commitment***. Generally, a random variable commitment scheme (RVCS) allows a prover to convince a verifier that a random variable was sampled according to some agreed-upon distribution $\boldsymbol{\mathcal{Z}}$, without learning anything else about the value of the random variable. In more detail, in an RVCS, the prover and verifier interact to produce a commitment $\mathbf{C}$ to a value $\mathbf{Z}$; at the end of the interaction, both parties are convinced that $\mathbf{Z} \leftarrow \boldsymbol{\mathcal{Z}}$, but only the prover knows the value of $\mathbf{Z}$.

In particular, we provide a construction for random bit commitments which gives a new modality of use for two-party coin-flipping. Coin flipping is a well-known primitive which was previously designed either for both parties to immediately learn the result of the flip or for one party to know the outcome and the other have delayed knowledge of the result (released later to show that the first party adhered to the protocol) [49, 18]. In our case, we need coin flipping where only one party ever learns the result and the other is immediately convinced of the unbiased nature of the flip without *ever* learning the result (it will only later learn certain functions of the result, which will not be used to verify the flip).

Efficient Protocol and Implementation of Certified DP. We complete the construction of Certified DP via the Binomial mechanism [36] by building an RVCS for the Binomial distribution. At a bird's-eye view, we take advantage of the additive structure of the Binomial mechanism, along with the additive structure of counting queries, to achieve an efficient CPM for differentially private release with very low overhead. Intuitively, the additive structure allows us to get away with a commitment scheme that supports additive homomorphism with perfectly hiding openings (without requiring a fully-homomorphic scheme), for which Pedersen commitments suffice. The resulting CPM allows us to streamline the implementation of our CPM into an interactive 3-round offline stage independent of queries and an non-interactive online stage with very low overhead for the analyst to certify queries.

We demonstrate the immediate feasibility of Certified DP in a prototype implementation of this Certified Binomial Mechanism for arbitrary counting queries. Concretely, certifying differential privacy only requires the analyst to open a single Pedersen commitment at query-time. In Section 5, we show that after a one-time setup phase and sampling query-independent randomness, clients using our prototype can complete each query by computing group operations that only scale linearly in the given predicate's sparsity s. Our support for arbitrary counting predicates yields expressive queries (e.g. counting individuals earning over an income threshold). At the same time, our prototype balances functionality with efficiency by allowing a limit to maximum predicate degree. We experimentally demonstrate that query time remains constant with respect to database size n, data dimension d, or privacy budget ϵ. For instance, we can efficiently query an $n = 7000$-record, $d = 37$-bit Census dataset [23] of age, sex, income, and education *in less than 2 ms.*

Structure of Manuscript. The remainder of the manuscript is structured as follows. The introduction continues with a technical overview of our contributions, followed by discussion of related works. In Section 2, we include some helpful preliminaries. Then, in Section 3, we present our new definitions for Certified Probabilistic Mechanisms and Certified DP and the connection between them. Next, in Section 4, we show a general approach for constructing certified additive noise probabilistic mechanisms using random variable commitments that can be cast within what we call the public registrar model. Finally, in Section 5 we instantiate our approach to certifiably release arbitrary-predicate counting queries via the DP Binomial mechanism, analyze the complexity of this scheme, and evaluate the performance of its implementation.

1.2 Technical Overview of Contributions

Our main technical contribution is to instantiate the CPM framework described above, to build a system for *Certified Differential Privacy.* We work in the setting of a single untrusted data curator, who agrees to release predicate counting queries of the form

$$f(D) = \sum_{x \in D} f(x)$$

for arbitrary predicate $f : \{0,1\}^d \to \{0,1\}$. As a first step, we formalize the goals of Certified DP. Our focus is on certifying that a fixed output from the curator satisfies DP. A bit more formally, we require three key properties.

- **Correctness**: *An honest analyst, who (interactively) queries an honest curator, receives an accurate and private release of the intended query.*
- **Honest-Curator DP**: *Every release of an honest curator satisfies DP.*
- **Dishonest-Curator DP**: *Whenever a curator cheats in a way that degrades the accuracy or privacy of the release, an honest analyst catches their deviation and raises a failure flag $\perp$ with probability proportional to this degradation.*

Importantly, instead of attempting to check the accuracy and DP guarantees after the fact, the Verifier will ensure that the Prover is releasing query answers based on an agreed-upon probabilistic mechansism $\mathbf{M}_f$, which is known to fulfill the given guarantees. Appealing to the more general framework, we show that implementing a CPM for a DP mechanism achieves the above Certified DP properties.

Theorem 1. *If $\mathbf{M}_f$ is a Differentially Private mechanism for releasing query f, then a CPM for $\mathbf{M}_f$ guarantees Certified DP.*

Since our definitions and protocols enjoy perfect correctness, in theory even one instance of $\perp$ is enough to conclude that the curator has engaged in malicious behavior, although we note that in practice, even if the curator is honest, some very small probability of $\perp$ can be happen due to computer error. One of the contributions of our work (Definition 3.1, Theorem 3.4) is to exactly characterize the amount of privacy degradation that is made possible by giving an allowance for unintentional error, so a $\perp$ toleration threshold can be chosen with this in mind. Once the proportion of $\perp$'s gets above this threshold, all further interaction with a curator should be halted and the curator should face some sort of censure, for either they have been negligent in their implementation or they have been actively malicious.

To achieve Certified DP, we turn to a classic (but relatively-less-popular) DP mechanism: the Binomial mechanism [36]. Originally studied in the context of distributed multi-party computation under DP, the Binomial mechanism enables DP release of statistics by adding a (private) sequence of random bits (and then finally subtracting the mean). In our setting, the central curator promises to respond to predicate counting queries with noisy answers of the form $f(D) + \mathbf{Z}$ for $\mathbf{Z}$, distributed as a Binomial RV with mean subtracted.

As discussed, we construct a CPM for the Binomial mechanism by carefully designing a random variable commitment scheme for the Binomial distribution, coupled homomorphically with an appropriate "functional" commitment scheme for opening predicate counting queries. Formally, for any additive noise mechanism, we can reduce the problem of constructing Certified DP to designing an RVCS for the noise distribution, satisfying an appropriate homomorphism with the set of query functions F.

Theorem 2. *For a class of functions F, let CS denote a commitment scheme to open $f(D)$ for $f \in F$. Suppose there exists a Random Variable Commitment scheme for $\boldsymbol{\mathcal{Z}}$ that supports additive homomorphism with CS. Then for any $f \in F$, there exists a CPM for $\mathbf{M}_f(D) = f(D) + \mathbf{Z}$.*

While our strategy can, in principle, be made to work for any additive noise mechanism, our focus on the Binomial mechanism allows us to reduce the problem further to flipping a single bit (since we already require additive homomorphism). A simple way to construct a random variable commitment for a bit flip is to perform a variant of the classic "coin flipping in the well" protocol [49, 18]. To get an RVCS, the prover commits to a coin flip, the verfier sends the prover their own coin, and the parties homomorphically evaluate the XOR of the coins.

The verifier additionally needs to ensure that the prover actually commited to a bit (and not another integer). Leveraging the Σ-protocols of [32, 64], the verifier can be convinced of this fact through a 3-round, public-coin protocol that maintains hiding for the prover's bit. Conveniently, we can run the Σ-protocols in parallel for all the bits that we need to flip, so the entire CPM only requires 3 rounds of interaction.

In order to support other common DP mechanisms, it is an interesting problem for future work to build efficient RVCS constructions for distributions such as discrete Gaussian and Laplace so that these can be slotted into our additive noise construction. With some care, existing discrete samplers such as those in [24] may be able to be efficiently implemented by applying homomorphism and Σ-protocols to our random bit commitments.

Stringing our components together, we obtain Certified DP via the Binomial mechanism for predicate counting queries. We make no restrictions on the predicates $f : \{0,1\}^d \rightarrow \{0,1\}$, though the cost of the protocols can be improved for structured predicates. We say a predicate f is s-sparse if the number of non-zero coefficients when representing f as a polynomial is upper bounded by s. With this in mind, we can state our main result.

Theorem 3. *Consider releasing Q predicate counting queries over a database D with n elements in $\{0,1\}^d$ under differential privacy. There exists a 3-round, public-coin Certified DP implementation of the Binomial Mechanism*[3] *to release predicate counting queries with the following costs.*

- *In the* ***offline preprocessing*** *stage, the protocol requires the database curator and verifier to perform $n \cdot 2^d$ parallel Σ-protocols once and $N = 8\log(2/\delta)/\epsilon^2$ single-bit RVCSs per query; finally each party performs $n \cdot 2^d + QN$ homomorphic additions.*
- *In the* ***online query*** *stage for an s-sparse query, the prover and verifier each non-interactively perform s homomorphic additions and scalar multiplications; the verifier opens 1 commitment.*

Note that the pre-processing work scales with 2^d to write down and validate a commitment for each different monomial over d-dimensional inputs. In other words, to support a restricted set of predicate queries where the number of monomials we need to reason about is bounded (e.g., low-degree queries), we can reduce the complexity of prover and verifier's up-front work. Naturally, other optimizations can be performed, for instance using a Merkle tree to succinctly commit to the sums of monomials, and then at query-time only revealing the monomial sums with non-zero coefficients.

The careful reader may note that our construction is a constant-round, public-coin interactive protocol; thus, it is tempting to try to remove interaction completely via the Fiat-Shamir transformation. While intuitively-appealing, we show that this strategy cannot be made to work: *the Fiat-Shamir transformation is inherently insecure for all (nontrivial) Certified Probabilistic Mechanisms.* This fact distinguishes our study of CPMs from traditional interactive proofs for formal languages and promise problems, motivating our use of a new model, which we call the public registrar.

Removing Interaction with a Public Registrar. An intriguing aspect of our CPM construction described above is that the work of the prover and verifier can be split into online work per query, preceeded by a more significant pre-processing that is used in all future queries. In our final contribution, we show how to leverage this split to give an implementation of the Certified DP mechanism that supports non-interactive queries. Specifically, we implement the CPM in a new three-party interaction that we call *the Public Registrar Model*, satisfying the following properties:

- **Input-hiding Registration:** The prover and a third-party *registrar* interact to "register" the CPM commitment. The prover's input and internal randomness remain hidden from the Registrar throughout the public-coin protocol.
- **Non-interactive querying:** After registration, the verifier issues queries to the prover and registrar, and can open the query based on the responses.

[3]Here, we give a guarantee of (ϵ, δ)-DP per query. Based on the number of queries Q, one can use the appropriate basic or advanced composition lemma to establish the necessary ϵ_0 to guarantee (ϵ, δ)-DP over the entire release.

The public registrar performs all of the interactive work up front, so that at query time, the analyst can issue queries non-interactively. Consequently, the prover and registrar can release queries to multiple analysts in parallel, without redoing work to certify the initial commitments. Notice that the input-hiding property protects the registrar from taking on liability by becoming a central point of attack for information about the databases of various data curators: the registrar never sees any data in the clear. In the context of the Binomial mechanism, the registrar never even learns the DP query values.

In all, we establish concrete bounds on the cost needed to implement the Certified DP implementation of the Binomial Mechanism.

Corollary 1. *There exists a Certified DP implementation of the Binomial Mechanism for counting queries in the public register model with the following costs:*

- ***Registration:*** *the prover and registrar do the offline preprocessing work scaling as* $n \cdot 2^d + QN$.
- ***Opening an s-sparse query:*** *the prover and registrar do s homomorphic additions and scalar multiplications, while the verifier opens* 1 *commitment.*

Implementation of Certified DP. We provide an implementation of this certified Binomial mechanism for arbitary counting queries and investigate how the construction practically scales as database size n, database dimension d, per-query privacy budget ϵ, and query polynomial sparsity s vary.[4] In Section 5, we show that after a comparatively large one-time setup phase and sampling query-independent randomness, verifiers using our prototype can complete queries by computing group exponentiations and multiplications that only scale linearly in s. We experimentally demonstrate that query time remains constant with respect to n, d, or ϵ. Thus, with a strong query privacy budget $\epsilon = 1$, we can efficiently query a $n = 7000$-record, $d = 37$-bit Census PUMS dataset of age, sex, income, and education to retrieve a differentially-private count of individuals with income over $\$262,144 = 2^{18}$ with sparsity $s = 63$. Total client query time is 1.9 ms, while verifying the query requires checking a single Pedersen commitment—38 μs. For efficiency, we support limiting the maximum degree of counting predicates. In this case, committing to all 2^{37} possible monomials would be intractable so we set a maximum degree of 6 (i.e. 2,324,784 predicates), balancing efficiency and functionality. To support further scaling, these commitments can also be performed in parallel. Our prototype is open-source and available at `https://github.com/jlwatson/certified-dp`.

1.3 Discussion and Related Works

Our study of Certified Probabilistic Mechanisms is in conversation with many ongoing areas of research, in the study of "trustworthy" data analysis methods, as well as classic topics within the field of cryptography. We conclude the introduction with discussion of related works and directions for further inquiry.

Certifying Differential Privacy. Prior works have established that, in generality, certifying that a given function satisfies DP is a challenging problem, both in a black-box and white-box

[4]We have implemented the protocol in the prover-verifier model, not the public registrar model.

analysis setting [47, 44]. To avoid these negative results, a number of papers provide DP certification techniques for restricted classes of functions [72, 43, 42, 10, 82, 6, 9]. In [66], Narayan, Feldman, Papadimitriou, and Haeberlen propose a system for "Verifiable DP" where the curator is trusted, but the analyst is untrusted. They develop a query language VFuzz to verify the analysts' behavior for accuracy and DP. As discussed, the work of Biswas and Cormode [15] is most similar to ours. For completeness, we include a direct comparison of the implementation of the systems, highlighting especially our efficiency at query time. See Section 6 for further detailed comparison.

Public Verifiability. One hope for certification in the public registrar model would be *public verifiability.* Towards this goal, we can imagine choosing the public coins used by the registrar based on the NIST randomness beacon [60] (after the prover has performed their commitment). Then any verifier can check for themselves that NIST's timestamped public coins were used to produce the private coins needed in the protocol. In particular, the verifier needn't have had any involvement in the original interaction. Thus we could get *publicly verifiable private coins* whose trustworthiness is backed by the NIST public randomness beacon.

DP in Theory and Practice. A variety of trust settings have been considered to achieve DP, including centralized [39], statistical local [59], computational local [71, 75], distributed DP [36], and the shuffle model [29]. Certified DP allows us to achieve the strong statistical properties of the centralized model, while removing blind trust in the curator. Additionally, we can certify the accuracy of the DP release, akin to distributed DP, but without relying on an MPC setup.

In practice, DP is now implemented in a number of SQL "engines" including [80, 58] from Google and Uber respectively, as well as [14] which is used by the IRS, Wikimedia Foundation, and US Census Bureau. These engines operate in the trusted-curator model. Incorporating certified DP into these engines could significantly strengthen their security and clients' trust.

Verifiable ML. Our work fits into the broader research program for certifying statistical claims made by a prover. Initiated by [30], a number of works now study interactive proofs for classic distribution testing problems [56, 55]. This inquiry has been extended to testing properties of machine learning algorithms [50, 65] and their training data [73, 27]. Another related line of work has been developing techniques for executing ML algorithms on homomorphically encrypted data [19, 46, 28, 61].

Functional Commitment Schemes for Low-Degree Polynomials. A key module of our framework for building CPMs for additive noise mechanisms requires the prover to commit to a collection of functional queries over the database. In spirit, this module is similar to the functional commitment schemes (FCS) for low-degree polynomials developed by [62, 63, 7, 26]. Due to differences in the intended setting, we need some properties not typically considered for FCSs but can also get away with weaker settings of certain parameters, such as succinctness.

1.3.1 Comparing Our Construction to Secure Two-Party Computation

Another alternative to our construction is to utilize secure multiparty computation (MPC) with dishonest players, or more specifically, secure two-party computation (2PC). This approach would capture $\mathbf{M}_f$ as a circuit with three inputs: the database D, the curator's randomness $\mathbf{R_P}$, and the analyst's randomness $\mathbf{R_V}$, so that $\mathsf{M}_f(D, \mathbf{R_P} \oplus \mathbf{R_V}) \overset{d}{=} \mathbf{M}_f(D)$ if at least one of $\mathbf{R_P}$ and $\mathbf{R_V}$ is drawn honestly. In this case, we can ensure that the curator does not learn the analyst's $\mathbf{R_V}$ ahead of time, so that they cannot bias the output via rejection sampling (as captured our cheating-prover soundness definition for CPMs in Section 1.1). However, there are significant downsides to this approach.

Since DP is currently used in practice without utilizing heavyweight cryptographic tools, we believe any certification proposal that could be realistically adopted needs to be as lightweight as possible. In particular, MPC would not efficiently support *publicly verifiable* queries. First, standard 2PC is no longer public coin for the analyst and cannot be cast within the public registrar model in order to make it publicly verifiable. Existing "publicly verifiable" MPC approaches provide certificates of misbehavior but do not give the randomness guarantees we require [11], so publicly verifiable DP as we define it is not achieved. Specifically, these approaches would publicly confirm that there *exists* $\mathbf{R_V}$ that gives the claimed output, but not that this value was chosen uniformly at random.

Thus in the case of 2PC, instead of the curator simply utilizing a trusted source of public coins to provide public verifiability, an *individual* analyst must be involved in the interaction and only they will be convinced of DP. This is a general roadblock because supporting public verifiability via a public-coin verifier inherently reveals their randomness to the prover as well. Finally, losing public verifiability has an additional effect in the case of DP, which is that the curator cannot reuse answers to the same query for multiple analysts, so they are forced to use up additional privacy budget to reanswer the query. Given the importance of public verifiability in the regulatory compliance settings we envision for certified DP, we therefore opted for an alternative CPM approach within the public registrar model.

2 Preliminaries

First, we will go through the notation and basic definitions we use for interactive protocols and probability distributions. Next, we present basic terms to describe probabilistic mechanisms for differential privacy. Finally, we recall the definition of a commitment scheme and the properties we will require in this work.

Interactive Protocol Notation. We consider interactive protocols between a Prover P and Verifier V with common input and private $\text{input}_{\mathsf{P}}$ and $\text{input}_{\mathsf{V}}$ respectively. Given such an interaction, we denote the output as $\textbf{Output} \leftarrow (\mathsf{P}(\text{input}_{\mathsf{P}}), \mathsf{V}(\text{input}_{\mathsf{V}}))(\text{input})$. Privately stored information and a party's view are denoted by $\textbf{Output}_{\mathsf{P}}$ and $\textbf{View}_{\mathsf{P}}$. We use typeface to distinguish between different probabilistic objects. For example, let randomized algorithm $\mathsf{A}(x) = x + \mathsf{Z}$ for random variable $\mathsf{Z} \leftarrow \mathcal{Z}$ with this probability distribution having support Z. We use $\lesssim$, $\gtrsim$, $\approx$, and $\overset{d}{\approx}$ to hide $\text{negl}(\lambda) = o(1/\text{poly}(\lambda))$ factors.

Distributional Closeness. We will use two notions of closeness between probability distributions. The first is based on standard statistical distance.

Definition 2.1 (Total Variation Distance)**.** Probability distributions $\mathcal{Z}_1$ and $\mathcal{Z}_2$ have *total variation distance* $TV(\mathcal{Z}_1, \mathcal{Z}_2) = \frac{1}{2}\sum_{z\in Z} |\Pr_{\mathcal{Z}_1}[z] - \Pr_{\mathcal{Z}_2}[z]|$. They are *$\delta$-statistically close* if $TV(\mathcal{Z}_1, \mathcal{Z}_2) \lesssim \delta$, denoted $\mathcal{Z}_1 \approx_{\delta\text{-TV}} \mathcal{Z}_2$. Otherwise they are *$\delta$-statistically far*, i.e. $\mathcal{Z}_1 \not\approx_{\delta\text{-TV}} \mathcal{Z}_2$.

The second notion of closeness is used to define differential privacy.

Definition 2.2 (Differential Closeness)**.** Probability distributions $\mathcal{Z}_1$ and $\mathcal{Z}_2$ are *(ϵ, δ)-differentially close* if for any event $E \subseteq Z$,

$$\Pr_{\mathcal{Z}_1}[E] \lesssim e^{\epsilon} \cdot \Pr_{\mathcal{Z}_2}[E] + \delta \text{ and } \Pr_{\mathcal{Z}_2}[E] \lesssim e^{\epsilon} \cdot \Pr_{\mathcal{Z}_1}[E] + \delta,$$

denoted $\mathcal{Z}_1 \approx_{(\epsilon,\delta)\text{-DP}} \mathcal{Z}_2$. Otherwise they are *$(\epsilon, \delta)$-differentially far*, i.e. $\mathcal{Z}_1 \not\approx_{(\epsilon,\delta)\text{-DP}} \mathcal{Z}_2$.

Probabilistic Mechanisms and Differential Privacy. A probabilistic mechanism is a randomized algorithm for computing a function. We are interested in probabilistic mechanisms which achieve certain definitions of accuracy and privacy.

Definition 2.3 (Accuracy)**.** Let function class $F \subseteq \{f : X^* \to Y\}$. Then an *$(\alpha, \beta)$-accurate probabilistic mechanism M_F for F* is a family of randomized algorithms $\{\mathsf{M}_f\}_{f\in F}$ with the following property: for all $f \in F$, $D \in X^*$, $\Pr[|f(D) - \mathsf{M}_f(D)| > \alpha] \lesssim \beta$.

Definition 2.4 (Differential Privacy)**.** Two databases $D, D' \in X^*$ are called *neighboring* if they differ on a single data point. A probabilistic mechanism M_F fulfills *(ϵ, δ)-differential privacy (DP)* if for all $f \in F$ and all neighboring databases D, $D' \in X^*$, $\mathsf{M}_f(D) \overset{d}{\approx}_{(\epsilon,\delta)\text{-DP}} \mathsf{M}_f(D')$. On the other hand, two neighboring databases $D, D' \in X^*$ *witness a violation of (ϵ, δ)-DP* if $\mathsf{M}_f(D) \overset{d}{\not\approx}_{(\epsilon,\delta)\text{-DP}} \mathsf{M}_f(D')$.

Definition 2.5 (Binomial Mechanism, [36, 4])**.** Let $\boldsymbol{\mathcal{B}}_N = \text{Binomial}(N, 1/2)$. The *Binomial additive noise mechanism* for F is given by $\mathsf{B}_f(D) = f(D) + \mathbf{B}_N - N/2$.

Theorem 2.1 ([36, 4])**.** *For functions f with sensitivity $\Delta(f) \leq 1$, if ϵ is sufficiently small and $N \geq 8\log(2/\delta)/\epsilon^2$, then B_f fulfills (ϵ, δ)-DP. By Hoeffding, for a given $\alpha > 0$ it achieves (α, β)-accuracy for $\beta = 2\exp(-2\alpha^2/N) = 2\exp(-\alpha^2\epsilon^2/4\log(2/\delta))$.*

Commitment Schemes and Homomorphism. A commitment scheme for the set of values X consists of three functions:

- **Setup** : security parameter $1^\lambda \to$ public parameters pp
- **Commit** : value $x \in X \to$ (commitment $\mathbf{C}_x$, decommitment/proof $\mathbf{\Pi}_x$)
- **Verify** : $(\mathbf{C}_x, \mathbf{\Pi}_x, x) \to x$ or $\perp$

We consider computationally-binding, perfectly-hiding commitment schemes (though a statistically-hiding scheme would also suffice).

Definition 2.6 (Commitment Scheme)**.** The functions (**Setup**, **Commit**, **Verify**) constitute a *commitment scheme* for value set X if the following three properties hold when $pp \leftarrow \textsf{Setup}(1^\lambda)$[5] and $(\mathbf{C}_x, \mathbf{\Pi}_x) \leftarrow \textsf{Commit}(x)$:

- *Correctness*: For any $x \in X$, $\textsf{Verify}(\mathbf{C}_x, \mathbf{\Pi}_x, x) = x$.
- *Computational Binding*: For any probabilistic polynomial-time (PPT) algorithm **A** such that $(\mathbf{C}, \mathbf{X}, \mathbf{\Pi}, \mathbf{X}', \mathbf{\Pi}') \leftarrow \mathsf{A}$,
$$\Pr\left[\mathbf{X} \neq \mathbf{X}' \wedge \textsf{Verify}(\mathbf{C}, \mathbf{\Pi}, \mathbf{X}) = \mathbf{X} \wedge \textsf{Verify}(\mathbf{C}, \mathbf{\Pi}', \mathbf{X}') = \mathbf{X}'\right] = \text{negl}(\lambda).$$
- *Perfect Hiding*: For any $x, x' \in X$, $\mathbf{C}_x \overset{d}{=} \mathbf{C}_{x'}$.

Definition 2.7 (Perfectly-Hiding Additive Homomorphism)**.** A commitment scheme supports *additive homomorphism* if there exists deterministic operation $\oplus$ such that $\mathbf{C}_{x_1+x_2} = \mathbf{C}_{x_1} \oplus \mathbf{C}_{x_2}$ and $\mathbf{\Pi}_{x_1+x_2} = \mathbf{\Pi}_{x_1} \oplus \mathbf{\Pi}_{x_2}$ fulfill correctness (for value $x_1 + x_2$) and hiding (note that binding automatically applies). Finally, homomorphic operations must preserve *perfectly hiding openings*: if $x_1 + x_2 = x_1' + x_2'$,
$$(\mathbf{C}_{x_1}, \mathbf{C}_{x_2}, \mathbf{C}_{x_1+x_2}, \mathbf{\Pi}_{x_1+x_2}) \overset{d}{=} (\mathbf{C}_{x_1'}, \mathbf{C}_{x_2'}, \mathbf{C}_{x_1'+x_2'}, \mathbf{\Pi}_{x_1+x_2'}).$$

Fact 2.2 (Scalar Homomorphism)**.** *An additively homomorphic scheme automatically supports scaling by a public value $a \in X$ via repeated addition, denoted by $\mathbf{C}_{a\cdot x} = a \otimes \mathbf{C}_x$ and $\mathbf{\Pi}_{a\cdot x} = a \otimes \mathbf{\Pi}_x$.*

For instance, Pedersen commitments support additive homomorphism with perfectly hiding openings [67]. For a commitment scheme to support other homomophic operations, analogous correctness and hiding requirements must hold.

[5]Note that we assume that all functions have implicit access to the public parameters pp and all probabilistic statements are also over the randomness used by **Setup**.

Σ-Protocols for Additively Homomorphic Commitments. We use a particularly structured and efficient type of protocol which gives us what are called witness-indistinguishable proofs of knowledge [32] for certain properties of additively homomorphic commitments. We will utilize these in Section 4.2 to construct a random bit commitment scheme.

Definition 2.8 (See e.g. [78])**.** A Σ-protocol is a kind of 3-round public-coin protocol which can be performed in parallel with other Σ-protocols and made non-interactive in the Random Oracle Model by applying the Fiat-Shamir transformation.

Definition 2.9. A protocol is a *proof of knowledge (PoK)* for relation R if it consists of an honest PPT Prover P and honest PPT Verifier V such that:

- *Perfect Completeness*: For any $(h, w) \in R$, $\Pr[1 \leftarrow (\mathsf{P}(w), \mathsf{V})(h)] = 1$.
- *Knowledge Soundness*: There exists a PPT knowledge extractor K such that for any PPT adversary $\widetilde{\mathsf{P}}$ with $\Pr[1 \leftarrow (\widetilde{\mathsf{P}}, \mathsf{V})(h)] > \text{negl}(\lambda)$, upon rewindable black-box access[6] to $\widetilde{\mathsf{P}}$ and input h, K outputs witness w with $\Pr[(h, w) \in R] = 1 - \text{negl}(\lambda)$.

Such a proof of knowledge is denoted by $\text{PoK}\,\{w \mid R(h, w)\}$.

Notice that knowledge soundness implies that for any PPT adversary $\widetilde{\mathsf{P}}$, on a given input h either

- the honest V interacting with $\widetilde{\mathsf{P}}$ rejects with probability $1 - \text{negl}(\lambda)$, or
- K can extract a valid witness w from $\widetilde{\mathsf{P}}$ with probability $1 - \text{negl}(\lambda)$.

Definition 2.10 ([41])**.** An interactive protocol for relation R is *perfect witness indistinguishable (WI)* if for any $(h, w), (h, w') \in R$ and adversary $\widetilde{\mathsf{V}}$, the distribution over transcripts generated by $(\mathsf{P}(w), \widetilde{\mathsf{V}})(h)$ is identical to that of $(\mathsf{P}(w'), \widetilde{\mathsf{V}})(h)$.

Theorem 2.3 ([32])**.** *Σ-protocols are perfect witness indistinguishable.*

Theorem 2.4 ([32, 64])**.** *For any additively homomorphic commitment scheme, there is a WI-PoK* $\{\mathbf{\Pi}_{\mathsf{DE}} \mid \mathit{Verify}(\mathbf{C}_{\mathsf{DE}}, \mathbf{\Pi}_{\mathsf{DE}}, 0) = 0 \;\vee\; \mathit{Verify}(\mathbf{C}_{\mathsf{DE}}, \mathbf{\Pi}_{\mathsf{DE}}, 1) = 1\}$*, which is specifically a Σ-protocol.*

See Appendix B for the concrete construction of this protocol.

[6]This means that K can run $\widetilde{\mathsf{P}}$ repeatedly with the random string of its choice.

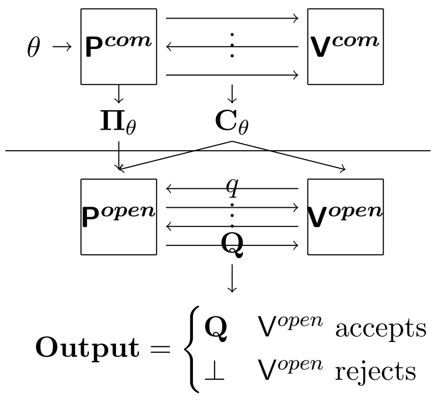

θ
P^{com}
V^{com}
$\mathbf{\Pi}_\theta$
$\mathbf{C}_\theta$
P^{open}
V^{open}
q
$\mathbf{Q}$
$\mathbf{Output} = \begin{cases} \mathbf{Q} & \mathsf{V}^{open} \text{ accepts} \\ \perp & \mathsf{V}^{open} \text{ rejects} \end{cases}$

Figure 1: Schematic of the Certified Probabilistic Mechanism Setup. A CPM is executed in two phases, the commitment and the query phase. The **Commitment Phase** is depicted in the top diagram: the Prover P^{com} receives a input θ, then interacts with the Verifier V^{com} to produce a commitment $\mathbf{C}_\theta$ and proof $\mathbf{\Pi}_\theta$. These outputs are passed to the bottom diagram depicting the **Query Phase**: the Verifier V^{open} sends a query q to the Prover P^{open}, then the two interact, during which the Prover sends their proposed query answer $\mathbf{Q}$. Based on the information they've received in both phases, the Verifier either accepts or rejects $\mathbf{Q}$ to produce the final **Output**.

3 Certified Probabilistic Mechanisms

Interactive Setting. In this paper, we will operate in an interactive protocol setting which allows the Verifier to hold the Prover to releasing queries q with respect to fixed, secret input θ. Analogously to a commitment scheme, we will have two phases: a **commitment phase** where the Prover commits to secret input θ of their choice, and a **querying phase** where the Prover releases answers based on the query q. Because the interaction is split into these phases, we use the following notation to disambiguate the roles of the Prover and Verifier in each phase, and in particular to restrict in which phase the parties may deviate from the protocol. See also Fig. 1.

Notation 3.1. Fix a Prover $\mathsf{P} = (\mathsf{P}^{com}, \mathsf{P}^{open})$ and a Verifier $\mathsf{V} = (\mathsf{V}^{com}, \mathsf{V}^{open})$ with inputs and outputs as follows:

- **Commitment Phase:** On input of $\theta \in \Theta$, P^{com} produces a commitment to D with V^{com}, privately storing the opening information: $(\mathbf{C}_\theta, (\mathbf{\Pi}_\theta)_{\mathsf{P}}) \leftarrow (\mathsf{P}^{com}(\theta), \mathsf{V}^{com})$.
- **Querying Phase:** On input of query $q \in Q$ from the Verifier and the saved outputs from the previous phase, P^{open} proposes an answer $\mathbf{Q}$ and V^{open} either accepts or rejects it: $\mathbf{Output} \leftarrow (\mathsf{P}^{open}(\mathbf{\Pi}_\theta), \mathsf{V}^{open})(\mathbf{C}_\theta, q)$ such that $\mathbf{Output} \in \{\mathbf{Q}, \perp\}$.

As in Definition 3.2, we will often denote the **Output** that results from running both phases in condensed form by $\mathbf{Output} \leftarrow (\mathsf{P}(\theta), \mathsf{V})(q)$ with the initial commitment phase being implicit.

In the definition to follow and the remainder of the main body of the paper, we will assume that the commitment phase is performed honestly. In Appendix A, we extend the definition (and our constructions) to handle a potentially dishonest commitment phase in a standard manner, analogously to requiring a proof of knowledge of the database. Additionally, we evaluate our implementation in the dishonest commitment setting in Section 5.

Definition of Certified Probabilistic Mechanisms. The goal of a "certified probabilistic mechanism" is for the Prover to convince the Verifier that they have answered according to $\mathsf{M}_q(\theta)$. If both parties are honest, they should simply produce a draw from $\mathsf{M}_q(\theta)$. If the Prover follows the intended protocol honestly, the Verifier should not be able to trick them into revealing unnecessary information about D or deviating from $\mathsf{M}_q(\theta)$. Whenever the Prover's release significantly deviates from $\mathsf{M}_q(\theta)$, the Verifier should catch them with correspondingly high probability.

Definition 3.1 (Certified Probabilistic Mechanism)**.** Given mechanisms $\mathsf{M}_q : \Theta \to Y$ for query class Q, a *certified probabilistic mechanism for* M_Q consists of Setup, an honest Prover $\mathsf{P} = (\mathsf{P}^{com}, \mathsf{P}^{open})$, and an honest Verifier $\mathsf{V} = (\mathsf{V}^{com}, \mathsf{V}^{open})$ with the following properties for public parameters $pp \leftarrow \mathsf{Setup}(1^\lambda)$, private database $\theta \in \Theta$, and public query $q \in Q$:

- *Correctness*: Let $\textbf{Output} \leftarrow (\mathsf{P}(\theta), \mathsf{V})(q)$. Then $\textbf{Output} \overset{d}{=} \mathsf{M}_q(\theta)$.
- *Cheating-Verifier Soundness*: For any adversary $\widetilde{\mathsf{V}}$, let $\textbf{Output} \leftarrow (\mathsf{P}(\theta), \widetilde{\mathsf{V}})(q)$ with proposed answer $\mathbf{Q}$. Then this is a perfectly hiding opening. Namely, for $y \in Y$ let $\textbf{View}^y_{\widetilde{\mathsf{V}}}(\theta)$ denote the view of $\widetilde{\mathsf{V}}$ in $(\mathsf{P}(\theta), \widetilde{\mathsf{V}})(q)$ conditioned on $\mathbf{Q} = y$. Then
$$\textbf{View}^y_{\widetilde{\mathsf{V}}}(\theta) \overset{d}{=} \textbf{View}^y_{\widetilde{\mathsf{V}}}(\theta').$$
Further, the distribution of the Prover's answer isn't distorted. Let $\mathbf{Q}_{\neq\perp}$ denote $\mathbf{Q}$ conditioned on the protocol not having terminated early. Then
$$\mathbf{Q}_{\neq\perp} \overset{d}{=} \mathsf{M}_q(\theta).$$
- *Cheating-Prover Soundness*: For any PPT adversary $\widetilde{\mathsf{P}}$, let $\textbf{Output} \leftarrow (\widetilde{\mathsf{P}}(\theta), \mathsf{V})(q)$ and $\gamma_\perp = \Pr[\textbf{Output} = \perp]$. Then
$$\textbf{Output} \overset{d}{\approx}_{\gamma_\perp\text{-TV}} \mathsf{M}_q(\theta).$$

Remark 3.1 (Statistical vs Computational)**.** Now suppose that the secret input is a database D, the query class is a function class F, and M_f is an (ϵ, δ)-DP mechanism. Notice that cheating-Prover soundness implies that the Prover is bound to some equivalence class of databases under M_F, namely those that give rise to the same outputs for all $f \in F$. If e.g. the mean of $\mathsf{M}_f(D)$, is slightly different for each database, the Prover is computationally bound to release from a unique database, while simultaneously maintaining statistical DP which hides neighboring databases. Depending on one's priorities, another version of a CPMs could be defined to require instead that any *unbounded* Prover fulfills *computational* DP (or gets caught). This is analogous to computationally binding and statistically hiding commitments vs vice versa.

On the face of it, cheating-Prover soundness only gives an inequality in the total variation distance between the output and intended mechanism in terms of the failure probability of the protocol. In fact, the property gives us a much stronger conclusion: all (but negligible) of the total variation distance is moved to the probability of catching the cheating Prover and outputting $\perp$.

Lemma 3.1. *Let* $\mathbf{M} \leftarrow \mathsf{M}_q(\theta)$ *and* $\mathbf{Output} \leftarrow (\widetilde{\mathsf{P}}(\theta), \mathsf{V})(q)$*. Then cheating-Prover soundness is equivalent to*

$$\Pr[\mathbf{Output} = \bot] = \sum_{y \in Y} (\Pr[\mathbf{M} = y] - \Pr[\mathbf{Output} = y]) \tag{1}$$

$$\textit{and for any } E \subseteq Y, \ \Pr[\mathbf{M} \in E] \gtrsim \Pr[\mathbf{Output} \in E]. \tag{2}$$

Proof sketch. Since M_q never outputs $\bot$, for **Output** to be distributed only TV-distance $\Pr[\mathbf{Output} = \bot] + \mathrm{negl}(\lambda)$ away, probabilities for all other outputs can only have decreased (modulo the $\mathrm{negl}(\lambda)$ factor). □

Since DP is defined in terms of events, we will also find a corollary which directly connects the probability of rejection to deviation from the mechanism for events E useful.

Corollary 3.1.1. *Let* $E \subseteq Y$*. Then*

$$\Pr[\mathbf{Output} = \bot] \gtrsim \Pr[\mathbf{M} \in E] - \Pr[\mathbf{Output} \in E], \tag{3}$$

$$\textit{and} \quad \Delta \gtrsim \Pr[\mathbf{M} \in E] - \Pr[\mathbf{Output} \in E]. \tag{4}$$

for $\Delta = TV(\mathsf{M}_q(\theta), (\widetilde{\mathsf{P}}(\theta), \mathsf{V})(q))$*.*

3.1 Certified Differential Privacy

Here, we introduce *Certified Differential Privacy.* Our definition aims to capture the following guarantee: whenever the Prover's release violates differential privacy or the promised accuracy, the honest Verifier will catch them and report the failure. On the other hand, when the Prover follows the protocol honestly, no Verifier can trick them into violating DP,[7] and if both parties are honest the output of the interaction satisfies the stated accuracy and privacy guarantees.

Definition 3.2 (Certified Differential Privacy)**.** A *certified* (α, β)*-accurate* (ϵ, δ)*-DP scheme* for function class $F \subseteq \{f : X^* \to Y\}$ consists of an honest Prover P and an honest Verifier V[8] with the following properties, for all databases $D \in X^*$ and functions $f \in F$:

- ***Correctness:*** The intended mechanism $\mathsf{M}_f(D) := (\mathsf{P}(D), \mathsf{V})(f)$ is (α, β)-accurate and (ϵ, δ)-DP with $\Pr[\mathsf{M}_f(D) = \bot] = 0$.
- ***Honest-Curator DP:*** For any (unbounded) adversarial verifier $\widetilde{\mathsf{V}}$, $\widetilde{\mathsf{M}}_f(D) := (\mathsf{P}(D), \widetilde{\mathsf{V}})(f)$ is an (ϵ, δ)-DP mechanism.
- ***Dishonest-Curator DP:*** For any PPT adversarial prover $\widetilde{\mathsf{P}}$, let

  $$\widetilde{\mathsf{M}}_f(D') := \begin{cases} (\widetilde{\mathsf{P}}(D'), \mathsf{V})(f) & D' = D \\ (\mathsf{P}(D'), \mathsf{V})(f) & \text{otherwise.} \end{cases}$$

  For any $\gamma \geq 0$, if the mechanism $\widetilde{\mathsf{M}}_f$ fails to satisfy $(\alpha, \beta + \gamma)$-accuracy or $(\epsilon, \delta + \gamma)$-DP, then $\Pr[\widetilde{\mathsf{M}}_f(D) = \bot] \gtrsim \kappa \cdot \gamma$, for some constant $\kappa \in (0, 1]$.

[7]This protection against a malicious Verifier is analogous to ZK (as compared to honest-Verifier ZK).

[8]Recall that in the CDP setting, the curator is a Prover and the analyst is a Verifier.

Remark 3.2 (Equivalent Formulation of Dishonest-Curator DP)**.** Dishonest-curator DP can be stated in the contrapositive, which will be more useful for constructions: Let $\gamma_\perp = \Pr[\widetilde{\mathbf{M}}_f(D) = \perp]$. Then, $\widetilde{\mathbf{M}}_f$ is an $(\alpha, \beta + c \cdot \gamma_\perp)$-accurate and $(\epsilon, \delta + c' \cdot \gamma_\perp)$-DP mechanism for f, for some $c, c' \geq 1$.

Remark 3.3 (Interpretation of Dishonest-Curator DP)**.** In a setting with a single database curator, the curator can always blatantly violate privacy (e.g., by outputting data in the clear). Dishonest-curator DP soundness requires that, *whenever* the curator leaks information (or fails to achieve the accuracy bound), the verifier will catch them (possibly with some constant factor degradation, as κ moves away from 1). Notice that if the Verifier outputs $\perp$ with probability $\gamma_\perp$, the best DP parameters we can hope for are $(\epsilon, \delta + \gamma_\perp)$. The increase in the DP failure probability is unavoidable, as outputting $\perp$ is itself disclosive, because the honest mechanism never outputs $\perp$. Similarly, outputting $\perp$ inherently degrades the mechanism's accuracy. Thus we require parameters $c, c' \geq 1$.

A particularly delicate aspect of the soundness property is why we define $\widetilde{\mathbf{M}}_f$ to distinguish between $D' = D$ and $D' \neq D$. While DP requires us to reason about behavior over all databases, in reality, the interaction is run only on the true database. Imagine a malicious Prover $\widetilde{\mathbf{P}}$ who follows the honest protocol on all but a single database D^*. On this special database, the Prover sends D^* in the clear and $\mathbf{V}$ outputs $\perp$. If $D \neq D^*$, then the Verifier's interaction with the Prover is as expected. But if we define $\widetilde{\mathbf{M}'}_f = (\widetilde{\mathbf{P}}(D'), \mathbf{V})(f)$ for all D', then $\widetilde{\mathbf{M}'}_f$ violates *any* DP guarantee, even though the Prover did nothing wrong in the real world. In this sense, considering $\widetilde{\mathbf{M}}_f$ allows us to localize our accuracy and DP guarantees based on deviations from the protocol—and privacy violations—that specifically occur on database D.

Remark 3.4 (Impact of Neighboring Databases)**.** Because DP is ultimately a local property, if we consider an adversary who releases a set of databases instead of only one, our definition automatically gives another guarantee due to either the worst-case behavior of the adversary on a single database in this set or the combined worst-case behavior on two neighboring databases in the set.

Lemma 3.2. *For any PPT adversary $\widetilde{\mathbf{P}}$ and set of databases $S \subseteq X^*$, let $N(S) = \{D_1, D_2 \in S \mid D_1, D_2$ are neighbors$\}$. Define*

$$\widetilde{\mathbf{M}}_f(D') := \begin{cases} (\widetilde{\mathbf{P}}(D'), \mathbf{V})(f) & D' \in S \\ (\mathbf{P}(D'), \mathbf{V})(f) & \textit{otherwise}, \end{cases}$$

and

$$\begin{aligned} \gamma_\perp^{\max} &= \max_{D \in S} \Pr[\widetilde{\mathbf{M}}_f(D) = \perp] \\ \gamma_\perp^{N\text{-}\max} &= \max_{(D_1, D_2) \in N(S)} \Pr[\widetilde{\mathbf{M}}_f(D_1) = \perp] + \Pr[\widetilde{\mathbf{M}}_f(D_2) = \perp]. \end{aligned}$$

Then $\widetilde{\mathbf{M}}_f$ is an $\big(\epsilon, \delta + O(\max\{\gamma_\perp^{\max}, \gamma_\perp^{N\text{-}\max}\})\big)$-DP probabilistic mechanism for f.

Our later constructions actually achieve an even stronger DP guarantee than this general guarantee for neighboring databases.

Remark 3.5 (Composition)**.** A key selling point of DP is its graceful degradation under composition [38, 40], and Certified DP maintains this. Formally, we give the following analogue of Basic Composition (as stated in [79]).

Lemma 3.3 (Composition of Certified DP)**.** *Consider any series of k adaptive queries $f_1, \dots, f_k \in F$. For honest prover P and any verifier $\widetilde{\mathsf{V}}$, the interactive mechanism is $(k\epsilon, k\delta)$-DP. Further, let*

$$\gamma_{\perp}^{sum} = \sum_{i \in [k]} \Pr[\widetilde{\mathsf{M}}_{f_i}(D) = \perp].$$

For any PPT cheating-prover $\widetilde{\mathsf{P}}$, the mechanism is $(k\epsilon, k\delta + O(\gamma_{\perp}^{sum}))$-DP.

Proof sketch. Let $\gamma_{\perp}^{i} = \Pr[\widetilde{\mathsf{M}}_{f_i}(D) = \perp]$. Since each M_{f_i} achieves $(\epsilon, \delta + O(\gamma_{\perp}^{i}))$-DP by dishonest-curator DP, we can apply the basic composition lemma (see e.g. [79]) to conclude the result. □

3.2 Certified DP from Certified Probabilistic Mechanisms

Now we are ready to prove that a certified probabilistic mechanism for a DP mechanism M_F attains certified DP. Notice that our dishonest-curator DP laxness constant e^{ϵ} gets closer to 1 the smaller ϵ is, i.e. we get a stronger certification guarantee for more stringent DP mechanisms. (As we will see later, this constant arises from the disconnect between DP-closeness and TV-closeness.)

Theorem 3.4 (restated Theorem 1)**.** *Let $\mathsf{M}_f : X^* \to Y$ be an (α, β)-accurate, (ϵ, δ)-DP mechanism for $f \in F$. Then a certified probabilistic mechanism CPM for M_F achieves certified (α, β)-accurate (ϵ, δ)-DP. Specifically, following the notation of Remark 3.2, CPM achieves $(\alpha, \beta + \gamma_{\perp})$-accuracy and $(\epsilon, \delta + e^{\epsilon} \cdot \gamma_{\perp})$-DP dishonest-curator soundness.*

Proof. **Correctness:** The certified DP correctness guarantee follows directly from correctness of a CPM applied to a (α, β)-accurate, (ϵ, δ)-DP mechanism.

Honest-Curator (ϵ, δ)-DP: We seek to establish that for any PPT adversary $\widetilde{\mathsf{V}} = (\widetilde{\mathsf{V}}^{com}, \widetilde{\mathsf{V}}^{open})$, their view across both phases of a verifiably DP scheme for $\mathbf{M}_F$ satisfies (ϵ, δ)-DP. We will consider the world where the protocol is run on D in comparison to the world where the protocol is run on D', named World D and World D' respectively. By cheating-Verifier soundness, for a fixed value of $\mathbf{Q} = y = \mathbf{Q}'$, the view of $\widetilde{\mathsf{V}}$ in World D is the same as that in World D', so the probability $\widetilde{\mathsf{V}}$ accepts vs rejects is also the same in both cases. Further, this means that $\widetilde{\mathsf{V}}$ had equal chance of misbehaving so as to cause P to answer $\perp$ in both worlds, so since

$$\mathbf{Q}_{\neq\perp} \overset{d}{=} \mathsf{M}_f(D) \overset{d}{\approx}_{(\epsilon,\delta)\text{-DP}} \mathsf{M}_f(D') \overset{d}{=} \mathbf{Q}'_{\neq\perp}$$

as well, the answers given by P maintain an (ϵ, δ)-DP mechanism overall. Thus

$$\left(\mathbf{View}_{\widetilde{\mathsf{V}}}(D), \mathbf{Q}\right) \overset{d}{\approx}_{(\epsilon,\delta)\text{-DP}} \left(\mathbf{View}_{\widetilde{\mathsf{V}}}(D'), \mathbf{Q}'\right),$$

and $\widetilde{\mathsf{V}}$'s entire view is (ϵ, δ)-DP, as desired.

Dishonest-Curator $(\alpha, \beta + \gamma_\perp)$-Accuracy: Let $\mathbf{Output} \leftarrow (\widetilde{\mathsf{P}}(D), \mathsf{V})(f)$ and $\mathbf{M} \leftarrow \mathsf{M}_f(D)$. Let $E = \{y \in Y \mid |f(D) - y| > \alpha\}$. Notice that $(\alpha, \beta + \gamma_\perp)$-accuracy is equivalent to

$$\Pr[\mathbf{Output} \in E] + \Pr[\mathbf{Output} = \perp] \leq \beta + \gamma_\perp + \text{negl}(\lambda).$$

In particular note that by convention we take $\perp$ to always be inaccurate, namely for any y and α, since we consider $|y - \perp| > \alpha$. By definition, $\Pr[\mathbf{Output} = \perp] = \gamma_\perp$. By Lemma 3.1,

$$\Pr[\mathbf{Output} \in E] \leq \Pr[\mathbf{M} \in E] + \text{negl}(\lambda) \leq \beta + \text{negl}(\lambda)$$

since M_f is (α, β)-accurate.

Dishonest-Curator $(\epsilon, \delta + e^\epsilon \cdot \gamma_\perp)$-DP: We establish this claim by Theorem 3.5. Further, this theorem shows that for a potentially misbehaving Prover who releases neighbors D_1, D_2, CPM is $(\epsilon, \delta + e^\epsilon \cdot \gamma_\perp^{\max})$-DP. □

This means that we get a stronger result than in Lemma 3.2. Namely, instead of the DP guarantee also degrading by the maximum sum of the rejection probabilities for neighboring $D_1, D_2 \in S$, it degrades only by the maximum rejection probability of a single database. This will be established via the following theorem, which is strictly more general than dishonest-curator DP since $\widetilde{\mathsf{P}}$ may simply follow P on one of the two databases.

Theorem 3.5. *Let $(\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ be a CPM for the (ϵ, δ)-DP mechanism M_F. For any PPT adversary $\widetilde{\mathsf{P}}$, if neighboring databases $D, D' \in X^*$ witness an $(\epsilon, \delta + \delta')$-DP violation for $(\widetilde{\mathsf{P}}(\cdot), \mathsf{V})(f)$, then either*

$$\Pr[(\widetilde{\mathsf{P}}(D), \mathsf{V})(f) = \perp] \gtrsim e^{-\epsilon}\delta' \text{ or } \Pr[(\widetilde{\mathsf{P}}(D'), \mathsf{V})(f) = \perp] \gtrsim e^{-\epsilon}\delta'.$$

Proof. Let $\mathbf{M} \leftarrow \mathsf{M}_f(D)$, $\mathbf{M}' \leftarrow \mathsf{M}_f(D')$, $\mathbf{Output} \leftarrow (\widetilde{\mathsf{P}}(D), \mathsf{V})(f)$, $\mathbf{Output}' \leftarrow (\widetilde{\mathsf{P}}(D'), \mathsf{V})(f)$, $\Delta = TV(\mathsf{M}_f(D), (\widetilde{\mathsf{P}}(D), \mathsf{V})(f))$, and $\Delta' = TV(\mathsf{M}_f(D'), (\widetilde{\mathsf{P}}(D'), \mathsf{V})(f))$. The $(\epsilon, \delta + \delta')$-DP violation means that for some event $E \subseteq Y \cup \{\perp\}$, WLOG

$$\Pr[\mathbf{Output} \in E] > e^\epsilon \cdot \Pr[\mathbf{Output}' \in E] + \delta + \delta'. \tag{0}$$

(The roles of D and D' could also be switched for this DP violation, in which case we would end up lower bounding the probability of rejection for the opposite database.) Recall that by cheating-Prover soundness, $\Pr[\mathbf{Output} = \perp] \gtrsim \Delta$ and $\Pr[\mathbf{Output}' = \perp] \gtrsim \Delta'$, so it suffices to show that at least one of the protocols is $e^{-\epsilon}\delta'$-statistically far from the correct mechanism. There are two cases.

First, suppose that $\perp \notin E$. We will show that $\Delta' > e^{-\epsilon}\delta'$. Overall, $\mathbf{M}'$ and $\mathbf{M}$ are (ϵ, δ)-DP close. For all events E such that $\perp \notin E$, $\Pr[\mathbf{M} \in E] \geq \Pr[\mathbf{Output} \in E] - \text{negl}(\lambda)$ due to Lemma 3.1. But by the assumed DP violation in (0), $\Pr[\mathbf{Output}' \in E]$ is significantly smaller than $\Pr[\mathbf{Output} \in E]$. We can thereby conclude that $\mathbf{M}'$ and $\mathbf{Output}'$ are also significantly far apart for this event. Specifically,

$$\begin{aligned} e^\epsilon \cdot \Pr[\mathbf{M}' \in E] + \delta &\geq \Pr[\mathbf{M} \in E] & (1)\\ &\geq \Pr[\mathbf{Output} \in E] - \text{negl}(\lambda) & (2)\\ &> e^\epsilon \cdot \Pr[\mathbf{Output}' \in E] + \delta + \delta' - \text{negl}(\lambda) & (3) \end{aligned}$$

(1) follows from $\mathbf{M} \overset{d}{\approx}_{(\epsilon,\delta)\text{-DP}} \mathbf{M}'$. (2) is due to Lemma 3.1 Eq. (2). (3) follows from (O). Now rearranging and applying Corollary 3.1.1 Eq. (4), we achieve

$$\Delta' \geq \Pr[\mathbf{M}' \in E] - \Pr[\mathbf{Output}' \in E] - \text{negl}(\lambda) > e^{-\epsilon}\delta' - \text{negl}(\lambda).$$

Second, suppose that $\perp \in E$. We will show that $\Pr[\mathbf{Output} = \perp] > e^{-\epsilon}\delta' - \text{negl}(\lambda)$ directly. In this case, we show that $\Pr[\mathbf{Output} \in E]$ must be significantly higher than $\Pr[\mathbf{M} \in E]$. But by Lemma 3.1 $\perp$ is the only element that can have higher probability for **Output** than **M**, so we must have a high probability of **M** equalling $\perp$. To reason about this setting, we split the event E into $E \setminus \{\perp\}$ and $\{\perp\}$ so that we can apply Lemma 3.1 and Corollary 3.1.1.

$$\Pr[\mathbf{Output} \in E \setminus \{\perp\}] + \Pr[\mathbf{Output} = \perp] \leq \Pr[\mathbf{M} \in E \setminus \{\perp\}] + \text{negl}(\lambda) + \Pr[\mathbf{Output} = \perp]$$

by Lemma 3.1 Eq. (2). Additionally, by rearranging Corollary 3.1.1 Eq. (3)

$$\Pr[\mathbf{Output}' \in E \setminus \{\perp\}] + \Pr[\mathbf{Output}' = \perp] \geq \Pr[\mathbf{M}' \in E \setminus \{\perp\}] - \text{negl}(\lambda).$$

Substituting these inequalities for $\Pr[\mathbf{Output} \in E]$ and $\Pr[\mathbf{Output}' \in E]$ respectively into the DP violation (0),

$$\begin{aligned}\Pr[\mathbf{M} \in E \setminus \{\perp\}] + \Pr[\mathbf{Output} = \perp] + \text{negl}(\lambda) &> e^{\epsilon} \cdot (\Pr[\mathbf{M}' \in E \setminus \{\perp\}] - \text{negl}(\lambda)) + \delta + \delta' \\ &\geq \Pr[\mathbf{M} \in E \setminus \{\perp\}] + \delta' - \text{negl}(\lambda)\end{aligned}$$

where the final inequality follows from the fact that $\mathbf{M} \overset{d}{\approx}_{(\epsilon,\delta)\text{-DP}} \mathbf{M}'$. Cancelling terms one more time, we conclude that $\Pr[\mathbf{Output} = \perp] > \delta' - \text{negl}(\lambda)$. □

Recall that in the definition of CDP, we required that a cheating Prover who causes an $(\epsilon, \delta+\gamma)$-DP violation results in $\Pr[\mathbf{Output} = \perp] \gtrsim \kappa \cdot \gamma$ for some $\gamma \in (0, 1]$. Our result in Theorem 3.5 shows that $\kappa \gtrsim e^{-\epsilon}$. We show that this analysis is actually tight. That is, due to the interplay between TV-closeness and differential closeness, an $e^{-\epsilon}$ loss is necessary. In particular, a small deviation in TV distance can get amplified into a larger DP violation by getting multiplied by e^{ϵ}, namely a Δ-deviation leading to $\delta' = e^{\epsilon} \cdot \Delta$ getting added to the DP violation. Suppose that E is an event which shows the tightness of the original (ϵ, δ)-DP guarantee with $\Pr[\mathbf{M} \in E] > \Pr[\mathbf{M}' \in E]$ without loss of generality, i.e.

$$\Pr[\mathbf{M} \in E] = e^{\epsilon} \cdot \Pr[\mathbf{M}' \in E] + \delta.$$

Suppose the Prover deviates so that $\Pr[\mathbf{O}' \in E] = \Pr[\mathbf{M}' \in E] - \Delta$, making no other changes so $\Delta = TV(\mathbf{M}', \mathbf{O}')$. Then

$$\begin{aligned}\Pr[\mathbf{O} \in E] &= \Pr[\mathbf{M} \in E] \\ &= e^{\epsilon} \cdot \Pr[\mathbf{M}' \in E] + \delta \\ &= e^{\epsilon} \cdot \big(\Pr[\mathbf{O}' = \perp] + \Delta\big) + \delta \\ &= e^{\epsilon} \cdot \Pr[\mathbf{O}' = \perp] + \delta + e^{\epsilon} \cdot \Delta.\end{aligned}$$

Thus the resulting mechanism violates $(\epsilon, \delta + \delta')$-DP for anything less than $\delta' = e^{\epsilon} \cdot \Delta$. Since the probability of $\perp$ is only necessarily as large as $\Delta - \text{negl}(\lambda)$ for a CPM, the Verifier may only be able to catch this violation with probability $\Delta - \text{negl}(\lambda) = e^{-\epsilon}\delta' - \text{negl}(\lambda)$.

4 Certified Additive Noise via Random Variable Commitments

We are now ready to discuss a general recipe for certified "additive noise" probabilistic mechanisms which leverages homomorphic commitments as well as a new primitive called an RVCS that we will introduce shortly. Specifically, we consider additive noise mechanisms of the following form.

Definition 4.1. Let M_F be an probabilistic mechanism that can be expressed as

$$\mathsf{M}_f(D) = f(D) + \mathbf{Z}$$

for noise $\mathbf{Z} \leftarrow \boldsymbol{\mathcal{Z}}$, drawn independently of $f \in F$ from a known distribution $\boldsymbol{\mathcal{Z}}$. Then M_F is an *additive noise mechanism.*

Conveniently for us, many DP mechanisms are of exactly this form. Our goal is for the Prover to produce a commitment to the value $\mathsf{M}_f(D)$ so that the Prover can open it to release the noisy value—without ever revealing the true value or added noise individually. Of course, this commitment must be generated in a way which convinces the Verifier that $\mathsf{M}_f(D) \leftarrow f(D) + \boldsymbol{\mathcal{Z}}$.

Towards this end, we start with a commitment scheme which supports homomorphism for the function class F.

Definition 4.2. Let $CS = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verifiy})$ be a commitment scheme for X. Let $F \subseteq \{f : (X^d)^* \to X\}$. CS supports *F-homomorphism* if there exists a randomized function which generalizes from commitment to an element to a commitment to a database denoted as $(\mathbf{C}_D, \mathbf{\Pi}_D) = \mathsf{Commit}(D)$ and for each $f \in F$ there exists a deterministic operation such that $(\mathbf{C}_{f(D)}, \mathbf{\Pi}_{f(D)}) = (f(\mathbf{C}_D), f(\mathbf{\Pi}_D))$ fulfill CS-correctness for opening value $f(D)$. (Note that binding is automatically inherited from CS.) Finally, homomorphic operations must preserve *perfectly hiding openings*: if $f(D) = f(D')$, then

$$(\mathbf{C}_D, \mathbf{C}_{f(D)}, \mathbf{\Pi}_{f(D)}) \overset{d}{=} (\mathbf{C}_{D'}, \mathbf{C}_{f(D')}, \mathbf{\Pi}_{f(D')}).$$

With this ingredient, we have a way to verifiably produce $\mathbf{C}_{f(D)}$, but we still need a way to add on the noise. This is where the new primitive that we call a "random variable commitment scheme" comes in. This will allow the Prover to verifiably produce $\mathbf{C}_{\mathbf{Z}}$ so that the Verifier is convinced that $\mathbf{Z} \leftarrow \boldsymbol{\mathcal{Z}}$. Once we have these two components, all we need is for CS to be additively homomorphic so that these commitments can be added together to produce $\mathbf{C}_{\mathsf{M}_f(D)} = \mathbf{C}_{f(D)} \oplus \mathbf{C}_{\mathbf{Z}}$, as desired.

4.1 Random Variable Commitments

A new primitive we use for our scheme, which seems interesting in its own right, is a notion of "committing to random variable $\boldsymbol{\mathcal{Z}}$." A Dealer DE and a Player PL seek to produce a commitment $\mathbf{C}$ to an element in Z together with the following properties. The Dealer can open $\mathbf{C}$, and is bound to open it to a single value $\mathbf{Z}$. This value is hidden from the Player until $\mathbf{C}$ is opened. But the Dealer and the Player are both convinced that $\mathbf{C}$ is a commitment to a value drawn from $\boldsymbol{\mathcal{Z}}$.

Recall that our definition of an interactive protocol means that in this case, at some point in the interaction the Dealer will propose a commitment $\mathbf{C}^Q$ which the Player will either accept or reject to produce an output $\mathbf{C}^{out} = \mathbf{C}^Q$ or $\mathbf{C}^{out} = \perp$ respectively.

Definition 4.3 (Random Variable Commitment Scheme)**.** A *random variable commitment scheme for distribution $\boldsymbol{\mathcal{Z}}$* consists of a commitment scheme $CS = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$, an honest Dealer DE, and an honest Player PL with the following properties when $pp \leftarrow \mathsf{Setup}(1^\lambda)$:

- *Correctness*: For $(\mathbf{C}^{out}, (\mathbf{\Pi}^Q, \mathbf{Z}^Q)_{\mathsf{DE}}) \leftarrow (\mathsf{DE}, \mathsf{PL})$ and $\mathbf{Z} \leftarrow \boldsymbol{\mathcal{Z}}$,
$$(\mathbf{C}^{out}, \mathbf{\Pi}^Q) \overset{d}{=} \mathsf{Commit}(\mathbf{Z}).$$
- *Cheating-Player Distributional Soundness*: For any adversary $\widetilde{\mathsf{PL}}$, let $(\mathbf{C}^{out}, (\mathbf{\Pi}^Q, \mathbf{Z}^Q)_{\mathsf{DE}}) \leftarrow (\mathsf{DE}, \widetilde{\mathsf{PL}})$, $\mathbf{Z}^{out}_{\mathbf{C} \neq \perp} \leftarrow \mathsf{Verify}(\mathbf{C}^{out}, \mathbf{\Pi}^Q, \mathbf{Z}^Q)$ conditioned on $\mathbf{C}^{out} \neq \perp$, and $\mathbf{Z} \leftarrow \boldsymbol{\mathcal{Z}}$. Then
$$\mathbf{Z}^{out}_{\mathbf{C} \neq \perp} \overset{d}{=} \mathbf{Z}.$$
- *Cheating-Dealer Distributional Soundess*: There exists a PPT algorithm $\mathsf{RevealOpening}$ that, given the Dealer's $\mathbf{View}$ and rewindable black-box access to the Dealer, returns $\mathbf{\Pi}^Q$ and $\mathbf{Z}^Q$ such that the following holds. For any PPT adversary $\widetilde{\mathsf{DE}}$, let $(\mathbf{C}^{out}, \mathbf{View}_{\widetilde{\mathsf{DE}}}) \leftarrow (\widetilde{\mathsf{DE}}, \mathsf{PL})$ and $(\mathbf{\Pi}^Q, \mathbf{Z}^Q) \leftarrow \mathsf{RevealOpening}(\mathbf{View}_{\widetilde{\mathsf{DE}}})$. Let $\mathbf{Z}^{out} \leftarrow \mathsf{Verify}(\mathbf{C}^{out}, \mathbf{\Pi}^Q, \mathbf{Z}^Q)$ and let $\gamma_\perp = \Pr[\mathbf{C}^{out} = \perp]$. Then for $\mathbf{Z} \leftarrow \boldsymbol{\mathcal{Z}}$,
$$\mathbf{Z}^{out} \overset{d}{\approx}_{\gamma_\perp\text{-TV}} \mathbf{Z}.$$

Cheating-Player distributional soundness ensures that the commitment is honestly openable to a value drawn from $\boldsymbol{\mathcal{Z}}$ when $\mathbf{C}^{out} \neq \perp$ cases are ignored. Cheating-Dealer distributional soundess ensures that if the Dealer draws from a distribution other than $\boldsymbol{\mathcal{Z}}$, including via producing an unopenable commitment, the honest Player catches their deviation proportionally.

Remark 4.1 (Asymmetry Between the Player and Dealer)**.** The asymmetry between cheating-Player and cheating-Dealer distributional soundness is due to the asymmetry in information between the two parties.

First, note that the Dealer must be able to compute $\mathbf{Z}^{out}$ given their view so they can open the commitment later. So, consider the most naïve strategy for altering the distribution that the Dealer can implement: depending on the $\mathbf{Z}^{out}$ that *would* be generated by an honest interaction, the Dealer sends $\mathbf{C}^Q = \perp$, so the Player is forced to output $\mathbf{C}^{out} = \perp$. In this way, they can decrease the probability that $\mathbf{Z}^{out} = z$ by whatever amount they want individually for each $z \in Z$—while paying for this by increasing the probability of $\perp$ by the same amount, of course. In particular, this allows the Dealer to alter the distribution even conditioning on $\mathbf{C}^{out} \neq \perp$. Cheating-Dealer distributional soundness says that the Dealer cannot do more than $\mathrm{negl}(\lambda)$ better than this naïve strategy.

On the other hand, cheating-Player distributional soundness implies that the Player is not able to behave differently depending on $\mathbf{Z}^{out}$, because then they could implement the same strategy as the Dealer to alter the conditional distribution. Thus $\mathbf{Z}^{out}$ must remain hidden to the Player.

In sum, the Dealer necessarily knows $\mathbf{Z}^{out}$ when generating the commitment, while this value is hidden from the Player. Thus a dishonest Player can only reject independently of the value of $\mathbf{Z}^{out}$, preserving the overall distribution. On the other hand, the Dealer can target specific values of $\mathbf{Z}^{out}$.

Remark 4.2 (Additive Homomorphism)**.** We will call an RVCS defined on top of a commitment scheme CS a random variable CS-commitment scheme. Note that if CS is additively homomorphic, then the RV CS-commitment scheme inherits this.

4.2 Random Bit Commitments Construction

The most simple instantiation of this primitive is to generate certified private random bits. Our construction builds on classical "coin flip in the well" constructions for cryptographic coin flipping [49, 18]. In these settings, two parties seek to generate a flipped coin that they can both trust was tossed with 50-50 probability of getting either bit. However, instead of requiring the flipped coin to be revealed later, we will use additive homomorphism to allow a commitment to the flipped coin to be generated without *ever* revealing its value to the Player.

We will use a couple of ingredients for this construction: (1) the witness-indistinguishable proof of knowledge that a commitment opens to either 0 or 1 from Theorem 2.4, and (2) a way to homomorphically XOR a bit commitment with a plaintext bit. The constructions for (1) and (2) are based on a standard Σ-protocol and CS's additive (and thus also scalar) homomorphism, and can be found in Appendix B.

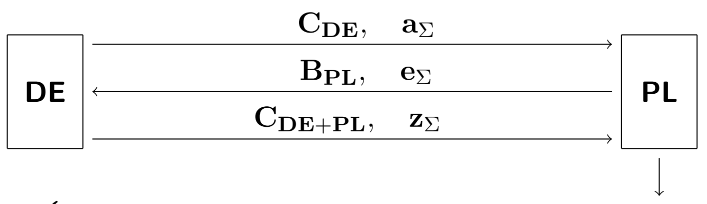
Construction 4.1. Let $CS = (\textsf{Setup}, \textsf{Commit}, \textsf{Verify})$ be a commitment scheme for $\mathbb{Z}/q\mathbb{Z}$ with additive homomorphism. Then we will define a $(\textsf{DE}_1, \textsf{PL}_1)$ protocol for $\boldsymbol{\mathcal{B}}_1$ as follows:

1. $\textsf{DE}_1$ flips a bit $\mathbf{B}_{\textsf{DE}} \leftarrow \boldsymbol{\mathcal{B}}_1$, computes a commitment to this bit $(\mathbf{C}_{\textsf{DE}}, \mathbf{\Pi}_{\textsf{DE}}) \leftarrow \textsf{Commit}$ $(\mathbf{B}_{\textsf{DE}})$, and sends $\mathbf{C}_{\textsf{DE}}$.
2. $\textsf{DE}_1$ and $\textsf{PL}_1$ execute WI-PoK $\{\mathbf{\Pi}_{\textsf{DE}} \mid \textsf{Verify}(\mathbf{C}_{\textsf{DE}}, \mathbf{\Pi}_{\textsf{DE}}, 0) = 0 \;\vee\; \textsf{Verify}(\mathbf{C}_{\textsf{DE}}, \mathbf{\Pi}_{\textsf{DE}}, 1) = 1\}$.
3. $\textsf{PL}_1$ flips a bit $\mathbf{B}_{\textsf{PL}} \leftarrow \boldsymbol{\mathcal{B}}_1$ and sends it.
4. $\textsf{DE}_1$ uses additive homomorphism to XOR $\mathbf{C}_{\textsf{DE}}$ and $\mathbf{\Pi}_{\textsf{DE}}$ with $\mathbf{B}_{\textsf{PL}}$ and sends the resulting commitment $\mathbf{C}_{\textsf{DE+PL}}$.
5. $\textsf{PL}_1$ uses additive homomorphism to XOR $\mathbf{C}_{\textsf{DE}}$ with $\mathbf{B}_{\textsf{PL}}$ and accepts iff the result matches $\mathbf{C}_{\textsf{DE+PL}}$.

At the end of the protocol, the Dealer knows the random bit $\mathbf{B}_{\textsf{DE+PL}}$ and the proof to open its commitment $\mathbf{\Pi}_{\textsf{DE+PL}}$. On the other hand, since the value $\mathbf{B}_{\textsf{DE}}$ has been hidden throughout, the Player only knows the commitment $\mathbf{C}_{\textsf{DE+PL}}$.

Remark 4.3 (Instantiation of Step #2)**.** If the Player doesn't confirm that $\mathbf{B}_{\textsf{DE}} \in \{0, 1\}$, then the supposed "bit" commitment produced at the end could be to a much larger element of $\mathbb{Z}/q\mathbb{Z}$. As highlighted in Section 2, additively homomorphic commitment schemes have efficient WI-PoKs of this kind called Σ-protocols.[9] While a naïve implementation of the above would result in a protocol with 6 rounds of interaction, we can reduce this back to 3. Specifically, since each message in the first round of the Σ-protocol is a perfectly hiding commitment, it can be implemented in parallel to the main interaction rounds in Construction 4.1, as depicted in Fig. 2, so it remains a 3-round (public-coin) protocol. Indeed, if many random bit commitments are needed, each can be performed in parallel.

[9]For commitment schemes with multiplicative homomorphism as well, or even just one-time multiplicative homomorphism (e.g. [45]), this additional homomorphism can be utilized to non-interactively check this instead.


$\mathbf{C_{DE}}, \quad \mathbf{a}_\Sigma$
$\mathbf{B_{PL}}, \quad \mathbf{e}_\Sigma$
$\mathbf{C_{DE+PL}}, \quad \mathbf{z}_\Sigma$
DE
PL

$$\mathbf{C}^{out} = \begin{cases} \mathbf{C_{DE+PL}} & \mathbf{C_{DE+PL}} = \mathbf{C_{DE}} + \mathsf{Commit}(\mathbf{B_{PL}}) \mod 2 \\ & \& \ \mathsf{Verify}_\Sigma(\mathbf{a}_\Sigma, \mathbf{e}_\Sigma, \mathbf{z}_\Sigma) = \text{accept} \\ \perp & \text{otherwise} \end{cases}$$

Figure 2: Diagram of the random bit commitment protocol, consisting of coin-flipping-in-the-well between the Dealer and Player with homomorphic commitments and a parallel Σ-protocol consisting of messages $(\mathbf{a}_\Sigma, \mathbf{e}_\Sigma, \mathbf{z}_\Sigma)$.

Theorem 4.1. *Let $CS = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$ be a commitment scheme for $\mathbb{Z}/q\mathbb{Z}$ with additive homomorphism. Let $(\mathsf{DE}_1, \mathsf{PL}_1)$ be from Construction 4.1 using CS. Then $(\mathsf{Setup}, \mathsf{DE}_1, \mathsf{PL}_1, \mathsf{Verify})$ constitute a random variable commitment scheme for $\boldsymbol{\mathcal{B}}_1$.*

Further, we can easily use this as a building block to get a protocol for Binomial commitments. Namely, we can simply execute Construction 4.1 multiple times in parallel and use additive homomorphism to add up the results.

Construction 4.2. Let $CS = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$ be a commitment scheme for $\mathbb{Z}/q\mathbb{Z}$ with additive homomorphism. Then we will define a $(\mathsf{DE}_N, \mathsf{PL}_N)$ protocol for $\boldsymbol{\mathcal{B}}_N$ as follows:

1. $(\mathsf{DE}_N, \mathsf{PL}_N)$ execute the protocol from Construction 4.1 N times in parallel, so that for $i \in [N]$ PL_N receives $\mathbf{C}_i$ and DE_N generates $\mathbf{\Pi}_i$ and the value $\mathbf{B}_i$.
2. PL_N homomorphically computes $\mathbf{C}_{\mathbf{B}_N} = \bigoplus_{i\in[N]} \mathbf{C}_i$. DE_N homomorphically computes $\mathbf{\Pi}_{\mathbf{B}_N} = \bigoplus_{i\in[N]} \mathbf{\Pi}_i$ and $\mathbf{B}_N = \sum_{i\in[N]} \mathbf{B}_i$.

Theorem 4.2. *Let $CS = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$ be a commitment scheme for $\mathbb{Z}/q\mathbb{Z}$ with additive homomorphism. Let $(\mathsf{DE}_N, \mathsf{PL}_N)$ be from Construction 4.2 using CS. Then $(\mathsf{Setup}, \mathsf{DE}_N, \mathsf{PL}_N, \mathsf{Verify})$ constitute a random variable commitment scheme for $\boldsymbol{\mathcal{B}}_N$.*

Proof. This follows directly from Theorem 4.1, the additive homomorphism of CS, and the perfectly hiding nature of the messages sent in the first round. □

This means that once we have random bit commitments in place, we will be prepared to release the Binomial additive noise mechanism.

By the definition of a group, we get the following lemma which will prove useful in the proof of Theorem 4.1.

Lemma 4.3. *Let $(G, \cdot)$ be a group and $\mathcal{G}$ denote the uniform distribution over elements in G. Then for $g \leftarrow \mathcal{G}$ and any independently chosen $g' \in G$, $g \cdot g'$ (and $g' \cdot g$) will be distributed according to $\mathcal{G}$ as well.*

Proof of Theorem 4.1. **Correctness:** The construction achieves RVCS correctness because the honest Dealer and Player conform to Lemma 4.3 for $\mathbb{Z}/2\mathbb{Z}$ so $\mathbf{C}_{\mathsf{DE+PL}} \overset{d}{=} \mathsf{Commit}(\mathbf{B}_1)$ for $\mathbf{B}_1 \leftarrow \boldsymbol{\mathcal{B}}_1$ and the Player accepts this result with probability 1. This is because correctness of CS's additive homomorphism implies that the honest Player will accept the honestly produced commitment $\mathbf{C}_{\mathsf{DE+PL}}$ with probability 1 and the Dealer will know the witnessing opening $(\boldsymbol{\Pi}_{\mathsf{DE+PL}}, \mathbf{B}_{\mathsf{DE+PL}})$, and perfect completeness of the PoK implies that the honest Player will accept the honest Dealer's PoK with probability 1.

Cheating-Player Distributional Soundness: This follows from the honest Dealer's choice of $\mathbf{B}_{\mathsf{DE}}$ being hidden from the Player so that the Player's choice of $\mathbf{B}_{\mathsf{PL}}$ is independent. Since the honest Dealer chooses a uniformly random bit and can see whether the Player chose an element of the group, i.e. $\mathbf{B}_{\mathsf{PL}} \in \{0,1\}$, we can apply Lemma 4.3. Since due to hiding and WI the Player never learns $\mathbf{B}_{\mathsf{DE}}$, their rejection probability is independent of $\mathbf{B}_{\mathsf{DE+PL}}$ and thus maintains distributional equality when conditioned on acceptance.

Let $\widetilde{\mathsf{PL}}$ be an PPT adversarial Player, $\mathbf{B}_1 \leftarrow \boldsymbol{\mathcal{B}}_1$, $(\mathbf{C}^{out}, (\boldsymbol{\Pi}^Q, \mathbf{B}^Q)) \leftarrow (\mathsf{DE}_1, \widetilde{\mathsf{PL}})$, $\mathbf{B}_{\mathbf{C}\neq\perp} = \mathsf{Verify}(\mathbf{C}^{out}, \boldsymbol{\Pi}^Q, \mathbf{B}^Q) \mid_{\mathbf{C}^{out}\neq\perp}$. Let $\mathbf{C}^Q$ denote the answer proposed by DE (so that $\mathbf{C}^{out} = \mathbf{C}^Q$ or $\mathbf{C}^{out} = \perp$ depending on whether $\widetilde{\mathsf{PL}}$ accepts or rejects). This implies that $\Pr[\mathbf{C}^{out} \neq \perp] > 0$, and we will restrict our attention to this case.

First, in this case certainly $\mathbf{C}^Q \neq \perp$, and $\mathbf{C}^{out} = \mathbf{C}^Q$. If $\widetilde{\mathsf{PL}}$ deviates from the structure of Construction 4.1 or in step #3 sends $\mathbf{B}_{\widetilde{\mathsf{PL}}} \notin \{0,1\}$, then in step #4, DE will set $\mathbf{C}^Q = \perp$, a contradiction. Thus $\mathbf{B}^Q \in \mathbb{Z}/2\mathbb{Z}$.

Next, we will establish independence between $\mathbf{B}_{\mathsf{DE}}$ and $\mathbf{B}_{\widetilde{\mathsf{PL}}}$. Let $\mathbf{B}^0_{\widetilde{\mathsf{PL}}}$ and $\mathbf{B}^1_{\widetilde{\mathsf{PL}}}$ indicate $\mathbf{B}_{\widetilde{\mathsf{PL}}}$ conditioned on $\mathbf{B}_{\mathsf{DE}} = 0$ and $\mathbf{B}_{\mathsf{DE}} = 1$ respectively, i.e. in World 0 versus World 1, and likewise with the other random variables in the protocol. In step #1, by perfect hiding $\mathbf{C}^0_{\mathsf{DE}} \overset{d}{=} \mathbf{C}^1_{\mathsf{DE}}$. In step #2, by perfect witness indistinguishability $\widetilde{\mathsf{PL}}$'s transcripts from the PoK are distributed identically in World 0 versus World 1. Thus $\widetilde{\mathsf{PL}}$'s entire transcript through the first three steps is identically distributed in Worlds 0 and 1, including $\mathbf{B}^0_{\widetilde{\mathsf{PL}}} \overset{d}{=} \mathbf{B}^1_{\widetilde{\mathsf{PL}}}$.

Therefore the requirements of Lemma 4.3 are fulfilled, so $\mathbf{B}^Q \overset{d}{=} \mathbf{B}_1$. By correctness of CS, $\mathsf{Verify}(\mathbf{C}^Q, \boldsymbol{\Pi}^Q, \mathbf{B}^Q) = \mathbf{B}^Q$. Further, since $\mathbf{C}^Q$ is also perfectly hiding, the probability of $\widetilde{\mathsf{PL}}$ rejecting remains the same in both Worlds 0 and 1. Thus $\mathbf{B}_{\mathbf{C}\neq\perp} \overset{d}{=} \mathbf{B}_1$.

Cheating-Dealer Distributional Soundness: This follows from the Player's choice of $\mathbf{B}_{\mathsf{PL}}$ happening after the Dealer's choice of $\mathbf{B}_{\mathsf{DE}}$ so the Dealer's choice is independent. Since the honest Player chooses a uniformly random bit and uses the WI-PoK to check that the Dealer chose an element of the group, i.e. $\mathbf{B}_{\mathsf{DE}} \in \{0,1\}$, we can apply Lemma 4.3. Because the Dealer does learn $\mathbf{B}_{\mathsf{PL}}$, their probability of misbehaving so as to cause rejection (e.g. sending $\mathbf{C}^Q = \perp$) may depend on $\mathbf{B}_{\mathsf{DE+PL}}$, which is why we are only able to ask for a weaker version of distributional soundness compared to when there is a cheating Player.

Let $(\mathbf{C}^{out}, \mathbf{View}_{\widetilde{\mathsf{DE}}}) \leftarrow (\widetilde{\mathsf{DE}}, \mathsf{PL}_1)$ and $\mathbf{C}^Q$ denote the answer proposed by $\widetilde{\mathsf{DE}}$ (so that $\mathbf{C}^{out} = \mathbf{C}^Q$ or $\mathbf{C}^{out} = \perp$ depending on whether PL accepts or rejects). Let K be the PPT knowledge extractor from the PoK in step #2. Let $\textsf{RevealOpening}$ compute $(\mathbf{\Pi}_{\widetilde{\textsf{DE}}}, \mathbf{B}_{\widetilde{\textsf{DE}}}) \leftarrow \textsf{K}(\mathbf{View}_{\widetilde{\textsf{DE}}})$ and follow step #4 to use these to compute $\mathbf{\Pi}^Q = \mathbf{\Pi}_{\widetilde{\textsf{DE}}+\textsf{PL}}$ and $\mathbf{B}^Q = \mathbf{B}_{\widetilde{\textsf{DE}}+\textsf{PL}}$, finally outputting $(\mathbf{\Pi}^Q, \mathbf{B}^Q)$. Let $\mathbf{B}^{out} = \textsf{Verify}(\mathbf{C}^{out}, \mathbf{\Pi}^Q, \mathbf{B}^Q)$. Our goal is to prove that

$$\mathbf{B}^{out} \overset{d}{\approx}_{\Pr[\mathbf{C}^{out}=\perp]\text{-TV}} \mathbf{B}_1$$

for $\mathbf{B}_1 \leftarrow \boldsymbol{\mathcal{B}}_1$, or equivalently,

$$\Pr[\mathbf{C}^{out} = \perp] \geq TV(\boldsymbol{\mathcal{B}}^{out}, \boldsymbol{\mathcal{B}}_1) - \text{negl}(\lambda)).$$

First, recall that by knowledge soundness of the PoK protocol as given by Definition 2.9, for this particular adversarial Dealer, either $\textsf{PL}$ rejects in step #2 with probability $1 - \text{negl}(\lambda)$ or $\textsf{K}$ can extract a valid witness with probability $1 - \text{negl}(\lambda)$. In the first case,

$$\Pr[\mathbf{C}^{out} = \perp] \geq 1 - \text{negl}(\lambda) \geq TV(\boldsymbol{\mathcal{B}}^{out}, \boldsymbol{\mathcal{B}}_1) - \text{negl}(\lambda),$$

as desired. So, will will turn our attention to the second case.

This means $(\mathbf{\Pi}_{\widetilde{\textsf{DE}}}, \mathbf{B}_{\widetilde{\textsf{DE}}})$ must be such that

$$\Pr[\textsf{Verify}(\mathbf{C}_{\widetilde{\textsf{DE}}}, \mathbf{\Pi}_{\widetilde{\textsf{DE}}}, \mathbf{B}_{\widetilde{\textsf{DE}}}) = \mathbf{B}_{\widetilde{\textsf{DE}}} \ \wedge \ \mathbf{B}_{\widetilde{\textsf{DE}}} \in \{0,1\}] \geq 1 - \text{negl}(\lambda).$$

Thus $\widetilde{\textsf{DE}}$ can open $\mathbf{C}_{\widetilde{\textsf{DE}}}$ to at least one of the values 0 or 1 with high probability. Suppose for contradiction that $\widetilde{\textsf{DE}}$ could also open $\mathbf{C}_{\widetilde{\textsf{DE}}}$ to a second value with non-negligible probability: then a PPT algorithm would be able to produce two conflicting openings of $\mathbf{C}_{\widetilde{\textsf{DE}}}$ which are accepted by $\textsf{Verify}$ with nonnegligible probability, which would contradict computation binding of CS. Thus there is only one value $\mathbf{C}_{\widetilde{\textsf{DE}}}$ can be opened to with nonnegligible probability of successful verification, so for a given realization of $\mathbf{C}_{\widetilde{\textsf{DE}}}$, $\mathbf{B}_{\widetilde{\textsf{DE}}}$ must be the same pseudodeterministic value for all but a negligible fraction of the times $\textsf{K}$ succeeds (even if different $\mathbf{\Pi}_{\widetilde{\textsf{DE}}}$'s may be produced).

Next, in step #3 $\textsf{PL}_1$ picks an independent $\mathbf{B}_{\textsf{PL}} \leftarrow \mathbf{B}_1$. Thus with probability $\geq 1 - \text{negl}(\lambda)$ Lemma 4.3 can be applied, leading us to conclude that $\mathbf{B}^Q \overset{d}{\approx}_{\text{negl}(\lambda)\text{-TV}} \mathbf{B}_1$. Further, for a correctly computed $\mathbf{C}_{\widetilde{\textsf{DE}}+\textsf{PL}}$, $\textsf{Verify}(\mathbf{C}_{\widetilde{\textsf{DE}}+\textsf{PL}}, \mathbf{\Pi}^Q, \mathbf{B}^Q) \overset{d}{\approx}_{\text{negl}(\lambda)\text{-TV}} \mathbf{B}_1$.

Now, of course $\widetilde{\textsf{DE}}$ may not send $\mathbf{C}^Q = \mathbf{C}_{\widetilde{\textsf{DE}}+\textsf{PL}}$, and instead send an incorrect commitment or even $\mathbf{C}^Q = \perp$. Since $\textsf{PL}_1$ deterministically checks whether $\mathbf{C}^Q = \mathbf{C}_{\widetilde{\textsf{DE}}+\textsf{PL}}$ and rejects if not, this probability of deviation translates directly into a probability of rejection. Let $p^0_{\mathbf{C}^Q=\perp}$ and $p^1_{\mathbf{C}^Q=\perp}$ indicate the probabilities that $\widetilde{\textsf{DE}}$ sends an incorrect $\mathbf{C}^Q$ and that $\mathbf{B}^Q = 0$ versus $\mathbf{B}^Q = 1$ respectively. Note that $\mathbf{C}^{out} = \perp$ in either of these cases, and otherwise $\mathbf{C}^{out} = \mathbf{C}^Q = \mathbf{C}_{\widetilde{\textsf{DE}}+\textsf{PL}}$. WLOG, let $p = p^1_{\mathbf{C}^Q=\perp} - p^0_{\mathbf{C}^Q=\perp} \geq 0$. Then

$$\mathbf{B}^{out} = \textsf{Verify}(\mathbf{C}^{out}, \mathbf{\Pi}^Q, \mathbf{B}^Q) \overset{d}{\approx}_{p\text{-TV}} \textsf{Verify}(\mathbf{C}_{\widetilde{\textsf{DE}}+\textsf{PL}}, \mathbf{\Pi}^Q, \mathbf{B}^Q) \overset{d}{\approx}_{\text{negl}(\lambda)\text{-TV}} \mathbf{B}_1.$$

Certainly, $p \leq \Pr[\mathbf{C}^{out} = \perp]$, so we can conclude that

$$\mathbf{B}^{out} \overset{d}{\approx}_{\Pr[\mathbf{C}^{out}=\perp]\text{-TV}} \mathbf{B}_1.$$

□

4.3 Certified Additive Noise Mechanisms Construction

Now supposing that we have an appropriate, additively homomorphic RV commitment scheme for $\boldsymbol{\mathcal{Z}}$, we'll see that this indeed gives us a certified probabilistic mechanism for $\mathsf{M}_f(D) = f(D) + \mathbf{Z}$. The Prover will use a F-homomorphic commitment scheme to produce a commitment to $f(D)$ and build the RVCS for $\boldsymbol{\mathcal{Z}}$ from this same CS. Choosing a CS with additive homomorphism will allow the commitments to $f(D)$ and $\mathbf{Z}$ to be combined, producing a commitment $\mathsf{M}_f(D)$ that can be opened while hiding both the true answer and the noise.

Construction 4.3. Let $CS = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$ be an F-homomorphic commitment scheme for X *with additive homomorphism.* Let $RVCS = (\mathsf{Setup}, \mathsf{DE}, \mathsf{PL})$ be a random variable CS-commitment scheme for $\boldsymbol{\mathcal{Z}}$. Then we define $\mathsf{P} = (\mathsf{P}^{com}, \mathsf{P}^{open})$ and $\mathsf{V} = (\mathsf{V}^{com}, \mathsf{V}^{open})$ as follows:

Commitment phase ($\mathsf{P}^{com}(D), \mathsf{V}^{com}$):

1. P^{com} computes $(\mathbf{C}_D, \mathbf{\Pi}_D) = \mathsf{Commit}(D)$ and sends $\mathbf{C}_D$ to V^{com}, privately storing $\mathbf{\Pi}_D$.

Randomness Generation Phase:

1. $RVCS$ is executed with P running DE and V running PL respectively, so that P^{open} saves $\mathbf{\Pi}_\mathbf{Z}$ and $\mathbf{Z}$ and V^{open} receives $\mathbf{C}_\mathbf{Z}$.

Querying Phase ($\mathsf{P}^{open}(\mathbf{\Pi}_D), \mathsf{V}^{open})(\mathbf{C}_D, f)$:

1. P^{open} F-homomorphically computes $\mathbf{\Pi}_{f(D)} = f(\mathbf{\Pi}_D)$ and $f(D)$. P^{open} additively homomorphically computes $\mathbf{\Pi}_{\mathbf{M}_f(D)} = \mathbf{\Pi}_{f(D)} \oplus \mathbf{\Pi}_\mathbf{Z}$ and $\mathbf{M}_f(D) = f(D) + \mathbf{Z}$ and sends both to V^{open}.

2. V^{open} F-homomorphically computes computes $\mathbf{C}_{f(D)} = f(\mathbf{C}_D)$. V^{open} additively homomorphically computes $\mathbf{C}_{\mathbf{M}_f(D)} = \mathbf{C}_{f(D)} \oplus \mathbf{C}_Z$. V^{open} outputs $\mathsf{Verify}(\mathbf{C}_{\mathbf{M}_f(D)}, \mathbf{\Pi}_{\mathbf{M}_f(D)}, \mathbf{M}_f(D))$.

Remark 4.4. Notice that the randomness generation phase must be run ahead time for each query but is independent of the specific query $f \in F$, so the querying phase itself becomes completely *non-interactive.* We will further explore the benefits of this in Section 4.4.

Theorem 4.4 (restated Theorem 2)**.** *Consider an additive noise mechanism $\mathsf{M}_f(D) = f(D) + \mathbf{Z}$ for class of functions $F \subseteq \{f : (X^d)^* \to X\}$ and $\mathbf{Z}$ drawn from distribution $\boldsymbol{\mathcal{Z}}$. Let $CS = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$ be an F-homomorphic commitment scheme for X* with additive homomorphism. *Let $RVCS = (\mathsf{Setup}, \mathsf{DE}, \mathsf{PL})$ be a random variable CS-commitment scheme for $\boldsymbol{\mathcal{Z}}$. Then Construction 4.3 is a certified probabilistic mechanism for M_F.*

Proof. **Correctness:** The construction achieves CPM correctness because $\mathbf{M}_f(D) \overset{d}{=} \mathsf{M}_f(D)$ and the Verifier accepts this result with probability 1. This is because correctness of $RVCS$ implies that the honest Verifier will accept the honestly produced $\mathbf{C}_\mathbf{Z}$ and the Prover will know the witnessing opening $(\mathbf{\Pi}_\mathbf{Z}, \mathbf{Z})$ with probability 1, and correctness of CS's F- and additive homomorphism implies that the honest Verifier will accept the honestly produced opening of $\mathbf{C}_{\mathbf{M}_f(D)}$ with probability 1.

Cheating-Verifier Soundness: This property follows from the perfectly hiding commitments and openings of CS (Definition 2.6, Definition 2.7) and cheating-Player distributional soundness for $RVCS$ (Definition 4.3) imply that that D remains hidden to the Verifier except through whatever is revealed by the value $\mathbf{M}_f(D)$ and that the Verifier is not able to alter the distribution of Z.

In detail, let $\widetilde{\mathsf{V}}$ be a PPT adversarial Verifier. Let $D, D' \in (X^d)^*$ with x_i and x_i' denoting elements of each respectively. We will consider the world where the protocol is run on D in comparison to the world where the protocol is run on D', named World D and World D' respectively. We seek to show that if the protocols in both worlds produce the same output $\mathbf{M}_f(D) = y = \mathbf{M}_f(D')$, then the views of the Verifier in both worlds are the same, namely

$$\mathbf{View}^y_{\widetilde{\mathsf{V}}}(D) \overset{d}{=} \mathbf{View}^y_{\widetilde{\mathsf{V}}}(D').$$

Notice that P does not take action based on anything sent by $\widetilde{\mathsf{V}}$ except when $RVCS$ is run. Thus up to that point P simply executes as normal and is unaffected by $\widetilde{\mathsf{V}}$ in either world. Notice then that before $RVCS$, P has only sent $\widetilde{\mathsf{V}}$ $\mathbf{C}_D$ in World D vs each $\mathbf{C}_{D'}$ in World D'. By perfect hiding of CS, $\mathbf{C}_D \overset{d}{=} \mathbf{C}_{D'}$. During the RVCS step, $\widetilde{\mathsf{V}}$'s view remains perfectly hiding between World D and World D' since DE sends messages identically in both worlds. Namely, these joint distributions are equal:

$$(\mathbf{C}_D, \mathbf{C}_{\mathsf{Z}}) \overset{d}{=} (\mathbf{C}_{D'}, \mathbf{C}_{\mathsf{Z}'}) \,.$$

This entails that the probability of $\widetilde{\mathsf{V}}$ outputting $\perp$ during the $RVCS$ step, i.e. that $\mathbf{C}_{\mathsf{Z}} = \perp$ or $\mathbf{C}'_{\mathsf{Z}} = \perp$ respectively, is also the same in both worlds. So, in the case that $\widetilde{\mathsf{V}}$ has already rejected in the $RVCS$ step, we're done.

Else if $\widetilde{\mathsf{V}}$ has not rejected yet, the only other messages $\widetilde{\mathsf{V}}$ receives are in the querying phase. One of these is the purported output of the mechanism. Conditioning on both worlds producing same output $\mathbf{M}_f(D) = y = \mathbf{M}_f(D')$, by perfectly hiding openings of CS

$$\left(\mathbf{C}_D, \mathbf{C}_{f(D)}, \mathbf{C}_{\mathsf{Z}}, \mathbf{\Pi}_{\mathbf{M}_f(D)}\right) \overset{d}{=} \left(\mathbf{C}_{D'}, \mathbf{C}_{f(D')}, \mathbf{C}_{\mathsf{Z}'}, \mathbf{\Pi}_{\mathbf{M}_f(D')}\right).$$

We have thereby shown that Construction 4.3 also has perfectly hiding openings. The second ingredient of cheating-Verifier soundness requires that the honest Prover's answer cannot be distributionally distorted by the Verifier, which is simple to show. Recall that by cheating-Player distributional soundness, $\mathsf{Z}^{out}_{\mathbf{C}\neq\perp} \overset{d}{=} \mathsf{Z}$. Thus $\mathbf{M}_f(D) = f(D) + \mathsf{Z}^{out}_{\mathbf{C}\neq\perp} \overset{d}{=} f(D) + \mathsf{Z} = \mathsf{M}_f(D)$, as desired.

Cheating-Prover Soundness: This property follows from the binding properties of CS (Definition 2.6) and cheating-Dealer distributional soundness for $RVCS$ (Definition 4.3) implying that the Prover deviating from M_F with some probability will cause the honest Verifier to reject with that same probability, up to negl(λ) factors.

Let $\widetilde{\mathsf{P}}$ be a PPT adversarial Prover, $\mathbf{M}^{out}_f \leftarrow (\widetilde{\mathsf{P}}(\mathbf{\Pi}_D), \mathsf{V}^{open})(\mathbf{C}_D, f)$, and $\mathbf{M}^Q_f$ denote the answer proposed by $\widetilde{\mathsf{P}}$, so that $\mathbf{M}^{out}_f = \mathbf{M}^Q_f$ or $\mathbf{M}^{out}_f = \perp$ depending on whether V accepts or rejects, respectively. Our goal is to prove that

$$\mathbf{M}^{out}_f \overset{d}{\approx}_{\Pr[\mathbf{M}^{out}_f = \perp]\text{-TV}} \mathsf{M}_f(D)$$

or equivalently,

$$\Pr[\mathbf{M}_f^{out} = \bot] \geq TV(\mathbf{M}_f^{out}, \mathsf{M}_f(D)) - \text{negl}(\lambda).$$

To establish this, let's trace the distributions of relevant RVs across the protocol. First, we need to examine randomness generation. Let $\mathbf{Z}^{out} = \textsf{Verify}(\mathbf{C_Z}, \mathbf{\Pi}_Z^Q, \mathbf{Z}^Q)$ for $(\mathbf{\Pi}_Z^Q, \mathbf{Z}^Q) \leftarrow \textsf{RevealOpening}$ $(\mathbf{View}_{\widetilde{\mathsf{P}}})$. By cheating-Dealer distributional soundness,

$$\mathbf{Z}^{out} \overset{d}{\approx}_{\Pr[\mathbf{C_Z}=\bot]\text{-TV}} \mathbf{Z} \tag{5}$$

for $\mathbf{Z} \sim \boldsymbol{\mathcal{Z}}$. Of course, if $\mathbf{C_Z} = \bot$ then V^{open} rejects and the protocol terminates. We will show that if $\mathbf{Z}^{out} = \bot$ with high probability, then by Eq. (5) $\mathbf{C_Z} = \bot$ with high probability so we will trivially achieve our goal. On the other hand, if $\mathbf{Z}^{out} \neq \bot$ with nonneglible probability, $\mathbf{C_Z}$ is openable to $\mathbf{Z}^Q$ with nonneglible probability. Thus by binding, $\mathbf{C_Z}$ (and by homomorphism $\mathbf{C}_{\mathbf{M}_{f(D)}}$) can be opened to this single value only. We will show that by Eq. (5), the distributional closeness of this unique value to the desired mechanism is bounded by the rejection probability, establishing the result. Next we will consider these two cases in detail.

In the first case, $\Pr[\mathbf{Z}^{out} = \bot] = 1 - \text{negl}(\lambda)$, so $TV(\mathbf{Z}^{out}, \mathbf{Z}) \geq 1 - \text{negl}(\lambda)$. Since Eq. (5) gives us an upper bound on this TV-distance, $\Pr[\mathbf{C_Z} = \bot] \geq 1 - \text{negl}(\lambda)$ as well. Since $\mathbf{C_Z} = \bot$ causes immediate rejection, then

$$\Pr[\mathbf{M}_f^{out} = \bot] \geq \Pr[\mathbf{C_Z} = \bot] \geq 1 - \text{negl}(\lambda) \geq TV(\mathbf{M}_f^{out}, \mathsf{M}_f(D)) - \text{negl}(\lambda)),$$

so we're done.

The second option is that $\Pr[\mathbf{Z}^{out} \neq \bot] = \Pr[\mathbf{Z}^{out} = \mathbf{Z}^Q]$ is nonnegligible. This means that $\widetilde{\mathsf{P}}$ *can* open $\mathbf{C_Z}$ to the value $\mathbf{Z}^Q$ with nonnegligible probability of success (using $\textsf{RevealOpening}$). Now let's move forward to the querying phase. Notice that by F-homomorphism, it is *possible* for $\widetilde{\mathsf{P}}$ to open $\mathbf{C}_{f(D)}$ to $f(D)$ since this is a well-formed commitment computed by V from the honest $\mathbf{C}_D$. Therefore $\widetilde{\mathsf{P}}$ can open both $\mathbf{C}_{f(D)}$ and $\mathbf{C_Z}$ to their respective values with nonnegligible probability, so by additive homomorphism $\widetilde{\mathsf{P}}$ *can* open $\mathbf{C}_{\mathbf{M}_{f(D)}}$ to $f(D) + \mathbf{Z}^Q$ with nonnegligible probability of success as well, so binding applies. Let p_{bad} denote the probability that $\widetilde{\mathsf{P}}$ gets to this step (i.e. the V didn't reject in the RVCS step i.e. $\mathbf{C_Z} \neq \bot$) and makes this attempt to open to a different value than $f(S) + \mathbf{Z}^Q$. Then

$$\mathbf{M}_f^{out} \overset{d}{\approx}_{p_{\text{bad}}\text{-TV}} f(D) + \mathbf{Z}^Q. \tag{6}$$

At the same time, by binding of CS, any attempt to open to a different value will get rejected with $1 - \text{negl}(\lambda)$ probability, so

$$\Pr[\mathbf{M}_f^{out} = \bot] \geq p_{\text{bad}} - \text{negl}(\lambda) + \Pr[\mathbf{C_Z} = \bot], \tag{7}$$

accounting as well for the case when V rejected earlier during the RVCS step. By Eq. (5), $\Pr[\mathbf{C_Z} = \bot]$ tells us how close $f(D) + \mathbf{Z}^Q$ is to the desired mechanism, namely

$$f(D) + \mathbf{Z}^Q \overset{d}{\approx}_{\Pr[\mathbf{C_Z}=\bot]\text{-TV}} f(D) + \mathbf{Z} = \mathsf{M}_f(D). \tag{8}$$

By the triangle inequality then, we can combine these three equations to get the desired result:

$$\mathbf{M}_f^{out} \overset{d}{\approx}_{p_{\text{bad}}\text{-TV}} f(D) + \mathbf{Z}^Q \overset{d}{\approx}_{\Pr[\mathbf{C_Z}=\bot]\text{-TV}} \mathsf{M}_f(D) \implies \mathbf{M}_f^{out} \overset{d}{\approx}_{\Pr[\mathbf{M}_f^{out}=\bot]\text{-TV}} \mathsf{M}_f(D).$$

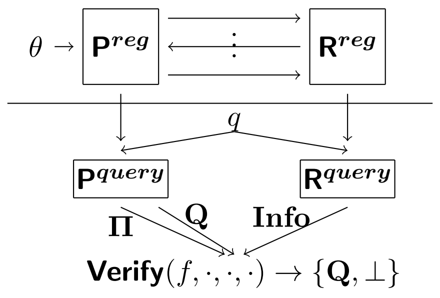


Figure 3: Diagram of the **Registration Phase** run on the Prover's secret input θ and **Query Phase** run on the query function $q \in Q$ where the Prover and Registrar non-interactively send proof and answer $(\mathbf{\Pi}, \mathbf{Q})$ and auxiliary **Info**, respectively, and the verification algorithm either accepts or rejects to produce the **Output**.

Namely, Eq. (6) and Eq. (8) imply that

$$TV(\mathbf{M}_f^{out}, \mathsf{M}_f(D)) \leq p_{bad} + \Pr[\mathbf{C_Z} = \perp] - \mathrm{negl}(\lambda) \leq \Pr[\mathbf{M}_f^{out} = \perp]$$

with this final inequality following from Eq. (7). □

4.4 Efficient Verification in the Public Registrar Model

Notice that in Construction 4.3, not only is the querying phase non-interactive, but before this step, everything is hidden in commitments so the database is completely indistinguishable for the Verifier. Further, the RVCS constructions we utilize are public-coin. Thus any specific instantiation of our CPM construction for additive noise mechanisms can operate within a specific interactive protocol structure we will call the "public Registrar model." Like CPMs, this model is more generally applicable than to only CDP, so we will use the more general Notation 3.1 for which there is a secret input $\theta \in \Theta$ and queries $q \in Q$.

This will allow the Verifier to offload the interactive components of the protocol to a trusted Registrar without the Registrar taking on liability by learning anything about the database D, or generally speaking the secret input θ—not even the DP query answers that the Verifier receives. Further, all of the interaction between the Prover and Registrar will occur during a registration phase *before* the query phase, so it can be done ahead of time. For Construction 4.3, this consists of the commitment and randomness generation phases. Thus the querying step becomes completely non-interactive between all parties. See Fig. 3.

Definition 4.4. An interactive protocol in the *public Registrar model* consists of an honest Prover $\mathsf{P} = (\mathsf{P}^{reg}, \mathsf{P}^{query})$, an honest public-coin Registrar $\mathsf{R} = (\mathsf{R}^{reg}, \mathsf{R}^{query})$, and a verification algorithm Verify with the following structure and properties for any input $\theta \in \Theta$ and query $q \in Q$:

- *Registration:* For input θ, we let $((\mathbf{\Pi}_\theta)_{\mathsf{P}}, (\mathbf{Info}_\theta)_{\mathsf{R}}) \leftarrow (\mathsf{P}^{reg}(\theta), \mathsf{R}^{reg})$.
- *Non-interactive querying:* The query phase is non-interactive such that $(\mathbf{\Pi}_{q(\theta)},\ \mathbf{Q}_{q(\theta)}) \leftarrow \mathsf{P}^{query}(\mathbf{\Pi}_\theta, q)$, $\mathbf{Info}_{q(\theta)} \leftarrow \mathsf{R}^{query}(\mathbf{Info}_\theta, q)$, and $\mathsf{Verify} : (q, \mathbf{Info}_{q(\theta)}, \mathbf{\Pi}_{q(\theta)}, \mathbf{Q}_{q(\theta)}) \rightarrow \{\mathbf{Q}_{q(\theta)}, \perp\}$.

- *Input indistinguishability:* For any adversary $\widetilde{\mathsf{R}}$, let $\mathbf{View}_{\widetilde{\mathsf{R}}}(\theta)$ be the view of $\widetilde{\mathsf{R}}$ from the protocol $(\mathsf{P}(\theta), \widetilde{\mathsf{R}})(q)$. Then for any inputs θ, θ',

$$\mathbf{View}_{\widetilde{\mathsf{R}}}(\theta) \overset{d}{=} \mathbf{View}_{\widetilde{\mathsf{R}}}(\theta').$$

Claim 4.1. Construction 4.3 can be implemented in the public Registrar model.

4.5 Insecurity of Removing Interaction via Fiat-Shamir

Given that the Registrar is public-coin, one might hope to go further and completely remove interaction in the Random Oracle Model using Fiat-Shamir. It turns out that this is not possible for *any* construction of certifiable probablistic mechanisms or DP. This is due to the fact that for these objects randomness is used not only for probabilistic *verification* of deterministic statements, but to produce probabilistic *outputs*. In the ROM, a cheating Prover can simply use rejection sampling on the honest protocol in order to distort the output distribution without getting caught.

Theorem 4.5. *Let $CPM = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ be a public-coin certifed probabilistic mechanism. Then the Fiat-Shamir of this protocol FS-CPM is insecure in the Random Oracle Model. Specifically, let $t(n)$ be the amount of time it take to run the honest protocol $(\mathsf{P}(D), \mathsf{V})(f)$. Then for any integer C and polynomial $p(n)$, there exists $\widetilde{\mathsf{P}}$ running in $C \cdot p(n) \cdot t(n)$ time such that*

$$\mathsf{Output} \overset{d}{\not\approx}_{(1-1/C)(1-1/p(n))\text{-}TV} \mathsf{M}_f(D),$$

but $\Pr[\mathsf{Output} = \perp] = 0$*, violating cheating-Prover soundness.*

Now, once again consider the classic "coin flipping in the well" problem. We can reconceptualize this as a certified probabilistic mechanism with no private database D and a single query which should lead to a value distributed as $\boldsymbol{\mathcal{B}}_1 := \text{Binomial}(1, 1/2)$. Suppose that the Verifier's random bit in this protocol is provided by a Random Oracle queried by the Prover so that the protocol becomes non-interactive. Now consider the following attack: the Prover will simply rerun the non-interactive protocol until the bit output at the end is 0.

There is a very similar, rejection sampling-based attack when applying Fiat-Shamir to certified probabilistic mechanisms in general, which we will utilize to establish Theorem 4.5 shortly. To get us started, we need a lemma first.

Lemma 4.6. *Let CPM be a public-coin certified probabilistic mechanism for M_Q. Let V' be a Verifier who sends bits when required by the protocol where each of these bits may be deterministically or randomly selected based on previous messages, and decides whether to accept at the end of the protocol by running the last step of V on the transcript. Let $\mathsf{M}'_q(\theta) := (\mathsf{P}(\theta), \mathsf{V}')(q)$. Then $\mathsf{M}'_q(\theta) \overset{d}{=} \mathsf{M}_q(\theta)$.*

Proof. This follows from correctness and cheating-Verifier querying soundness. By cheating-Verifier querying soundness,

$$\mathbf{Answer}_{\neq\perp} \overset{d}{=} \mathsf{M}_q(\theta).$$

Since the Verifier V' responds with a message that is equally likely to have been sent by the honest V in each step of the protocol, from P's view they are indistinguishable from V. Thus by correctness, P never sends a $\perp$ message to terminate the protocol early, so $\mathbf{Answer}_{\neq\perp} = \mathbf{Answer}$. Then again by correctness, when running V for the final step, V' will let

$$\mathbf{Output} = \mathbf{Answer} = \mathbf{Answer}_{\neq\perp} \overset{d}{=} \mathsf{M}_q(\theta),$$

as desired. □

Proof of Theorem 4.5. In short, by Lemma 4.6, the malicious Prover $\widetilde{\mathsf{P}}$ can simply rerun the honest Prover P until they get an output from M_q they like. This distorts the distribution proportionally to how many tries $\widetilde{\mathsf{P}}$ is willing to make.

Let $E_{p(n)} \subseteq Y$ such that $\Pr[\mathsf{M}_q(\theta) \subseteq E_{p(n)}] = \frac{1}{p(n)}$. Consider the following adversarial strategy for $\widetilde{\mathsf{P}}$: (1) Run $(\mathsf{P}(D), \mathcal{R})(q)$ $C \cdot p(n)$ times, generating some $\mathbf{Answer}_i$ each time. (2) If any of these runs have $\mathbf{Answer}_i \in E_{p(n)}$, send this transcript to the Verifier. Else send the first transcript.

By Lemma 4.6, even if the Random Oracle makes some of the Verifier's coins deterministic (because there is a repeated query from a previous run), $\mathbf{Output}_i \overset{d}{=} \mathsf{M}_q(\theta)$. Thus $\mathbf{Output}_i = \mathbf{Answer}_i$ with $\Pr[\mathbf{Output} = \perp] = 0$ and $\Pr[\mathbf{Output}_i \subseteq E_{p(n)}] = \frac{1}{p(n)}$ for each run. Thus the expected number of attempts $\mathbf{T}$ before success is $\mathbb{E}[\mathbf{T}] = p(n)$, so by Markov, the probability of failure with $C \cdot p(n)$ attempts is $\leq 1/C$. So since $\Pr[\exists i \in [C \cdot p(n)] \mid \mathbf{Output}_i \subseteq E_{p(n)}] \geq 1 - 1/C$ in which case the output is $(1 - 1/p(n))$-TV far from $\mathsf{M}_q(\theta)$ (and else the output is distributed the same as $\mathsf{M}_q(\theta)$), we find that

$$\mathsf{Output} \overset{d}{\not\approx}_{(1-1/C)(1-1/p(n))\text{-TV}} \mathsf{M}_q(\theta),$$

as desired. □

Thus we cannot hope to use Fiat-Shamir to make *any* certified randomized mechanisms or certified DP schemes non-interactive, even though we show in Section 4.4 that specific instances of each can be achieved in the public Registrar model.

Remark 4.5 (Insecurity of Fiat-Shamir for CDP)**.** Again quite similarly, there is an attack against the Fiat-Shamir of any random variable commitment scheme or certified DP protocol (whether or not the construction is based on a certified probabilistic mechanism).

Remark 4.6 (Minimal Interaction in Our Constructions)**.** Since we have just shown that interaction is necessary, and Construction 4.3 uses only 3 rounds of interaction, it can be considered minimally interactive. Specifically, in our construction all Σ-protocols can be securely Fiat-Shamired via standard methods (see Appendix B), but the coin-flipping components of the random bit commitments cannot. As remarked upon in Section 1.3, this component can be made publicly verifiable by utilizing a single random bit from the NIST public randomness beacon per random bit commitment.

5 Answering Counting Queries with Certified DP

Finally, we will instantiate Construction 4.3 in order to certifiably release DP answers for an expressive query class using the Binomial additive noise mechanism. Recall from Section 2 that the Binomial mechanism B_F [36, 4] is given by

$$\mathsf{B}_f(D) = f(D) + \mathbf{B}_N - N/2$$

for $\mathbf{B}_N \leftarrow \boldsymbol{\mathcal{B}}_N$ and achieves (ϵ, δ)-DP for functions f with sensitivity ≤ 1 when N is sufficiently large, along with an associated (α, β)-accuracy. As shown by [4], the Binomial mechanism achieves nearly the same accuracy-privacy tradeoff as the Gaussian mechanism while utilizing fewer representation bits, making it a natural choice for interactive settings.

To warm up, suppose we have data domain $X = \{0, 1\}$ so that our databases consist of 0-1 bits and that we seek to release the function class $F_{Sum} = \{f_S \mid f_S(D) = \sum_{i \in S} x_i \mod q\}$ for q a large prime. Notice that the sensitivity $\Delta(f_S) \leq 1$. It is straightforward to support releasing functions from F_{Sum} as long as we have a commitment scheme with additive homomorphism.

Construction 5.1. Let $CS = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$ be an additively homomorphic commitment scheme. Then CS supports F_{Sum}-homomorphism as follows:

- Let $\mathbf{C}_D$ and $\mathbf{\Pi}_D$ consist of a list of each $(\mathbf{C}_{x_i}, \mathbf{\Pi}_{x_i}) \leftarrow \mathsf{Commit}(x_i)$ for $x_i \in D$ respectively.
- Then for a given $f_S \in F_{Sum}$, $f_S(\mathbf{C}_D) = \bigoplus_{i \in S} \mathbf{C}_{x_i}$ and $f_S(\mathbf{\Pi}_D) = \bigoplus_{i \in S} \mathbf{\Pi}_{x_i}$.

Theorem 5.1. *If CS is a commitment scheme for $\mathbb{Z}/q\mathbb{Z}$ with additive homomorphism, then there exists a certified probabilistic mechanism for $\mathsf{B}_{F_{Sum}}$. In the CPM, at query time the Prover and Verifier each make $|S|$ homomorphic additions and the Verifier checks 1 commitment opening. By Theorem 3.5, this is also a certified (α, β)-accurate (ϵ, δ)-DP scheme for F_{Sum}.*

Proof sketch. This is achieved by instantiating Construction 4.3 for additive noise mechanisms with Construction 5.1 for F_{Sum} and Construction 4.2 for $\boldsymbol{\mathcal{B}}_N$ and applying Theorem 4.4. □

The function class F_{Sum} is somewhat artificial. More typically in DP settings, what are called "counting queries" are the primary interest. Instead of having a single bit associated with each individual and simply summing these, there is a vector associated with each individual which is then fed into a predicate that assigns the individual to 0 or 1. Formally,

Definition 5.1. Let $D \in (X^d)^*$ be a database with data domain $X = \{0, 1\}$ and $x_i \in X^d$ the ith member of the DB. Let $f : X^d \to \{0, 1\}$ be a predicate on these x_i. Then the *counting query* for f is defined as

$$f(D) = \sum_{i=1}^{n} f(x_i).$$

Then the class of counting queries with arbitrary predicates expressed as multilinear polynomials is defined as follows.

Definition 5.2. Let $x_{i,j}$ denote the jth bit in $x_i \in X^d$. Let $S \subseteq [d]$ and define the monomial $m_{i,S} = \Pi_{j \in S} x_{i,j}$. Then

$$F_{Count} = \left\{ \sum_{i=1}^{n} a_S \cdot m_{i,S} \;\middle|\; a_S \in \mathbb{Z}/q\mathbb{Z} \right\}.$$

For a given $f \in F_{Count}$, let M_f denote the set of monomials with nonzero a_S in f and $s_f = |M_f|$ denote the *sparsity* of f.

Note that the sensitivity of these counting queries is 1, as before. As argued in [17], this kind of query is a powerful primitive that captures a wide range of natural data analysis tasks in DP settings.

We can extend our previous construction for certified DP to support this function class F_{Count} as long as 2^d is not too large. As pointed out in [26], it is simple to homomorphically support "linearizable" functions, i.e. functions which can be expressed as linear functions of a small set of precomputed values. The immediate implementation of this is to commit to each monomial $m_{i,S}$, but given the form of Definition 5.2, by linearity it suffices to only commit to each monomial sum $m_S = \sum_{i \in [n]} m_{i,S}$ since for a given predicate each term in the sum only needs to be multiplied by the same public constant a_S.

Construction 5.2. Let $CS = (\textsf{Setup}, \textsf{Commit}, \textsf{Verify})$ be an additively homomorphic commitment scheme. Then CS supports F_{Count}-homomorphism as follows:

- Let $\mathbf{C}_D$ and similarly $\mathbf{\Pi}_D$ consist of each $\mathbf{C}_{m_S} = \bigoplus_{i \in [n]} \mathbf{C}_{m_{i,S}}$ and $\mathbf{\Pi}_{m_S} = \bigoplus_{i \in [n]} \mathbf{\Pi}_{m_{i,S}}$ respectively for $S \subseteq [d]$.
- Then for a given $f \in F_{Count}$,

$$\mathbf{C}_{f(D)} = \bigoplus_{S \in M_f} (a_S \otimes \mathbf{C}_{m_S}) \quad \text{and} \quad \mathbf{\Pi}_{f(D)} = \bigoplus_{S \in M_f} (a_S \otimes \mathbf{\Pi}_{m_S}).$$

Theorem 5.2 (restated Theorem 3)**.** *If CS is a commitment scheme for $\mathbb{Z}/q\mathbb{Z}$ with additive homomorphism, then there exists a certified probabilistic mechanism for $\mathbf{B}_{F_{Count}}$ based on CS. In the CPM, at query time the Prover and Verifier each make sparsity s_f homomorphic additions and multiplications and the Verifier checks 1 commitment opening. By Theorem 3.5, this is also a certified (α, β)-accurate (ϵ, δ)-DP scheme for F_{Count}.*

Proof sketch. This is achieved by instantiating Construction 4.3 for additive noise mechanisms with Construction 5.2 for F_{Count} and Construction 4.2 for $\boldsymbol{\mathcal{B}}_N$ and applying Theorem 4.4. □

See Appendix A for how to support a potentially dishonest commitment phase.[10]

[10]Also note that commitments can be concisely summarized using a vector commitment [25] to succinctly store each monomial commitment $\mathbf{C}_{m_S}$. Then when certain monomial sums are requested, the vector commitment can be opened to just these in particular. For our purposes, we would need the typical requirements of position-binding, succinctness, and efficient openings. Perhaps the most practical construction is to simply use a Merkle tree.

	# rounds	**P** #**Commit**	**V** #**Verify**	**P** #$\oplus$ & #$\otimes$	**V** #$\oplus$ & #$\otimes$
Commitment	1	$\lvert M\rvert \leq 2^d$	0	0	0
Random Bit Generation	3	3 per bit	2 per bit	2 per bit & 2 per bit	3 per bit & 3 per bit
Randomness N-Addition	0	0	0	$N-1$ & 0	$N-1$ & 0
Querying	1	0	1	s_f & s_f	s_f & s_f

Table 1: The number of rounds of interaction, commitments or verifications, and homomorphic additions and multiplications the Prover and the Verifier will have to do in each phase to carry out honest-commitment Certified DP for $\mathbf{B}_{F_{count}}$.

Complexity Analysis. In Table 1 above, we break down the number of rounds of interaction, commitment creations or verifications, and homomorphic additions and multiplications the Prover and the Verifier will have to do in each phase to carry out verifiable DP for $\mathbf{B}_{F_{count}}$. We let M denote the number of distinct monomials utilized by any of query predicates f and let s_f denote the number of monomials in the predicate, i.e. the sparcity of f. As discussed in Appendix A, a dishonest commitment phase ultimately involves $n \cdot M \leq n \cdot 2^d$ Σ-protocols which can be performed in parallel to support all of F_{count}. We expose the constants involved to make clear the practicality of this scheme. Recall that $N = 8\log(2/\delta)/\epsilon^2$ suffices. Typically ϵ is a constant ≤ 1 and δ is negligible in the size of the database. For instance, if $\epsilon = 1$ and $\delta = 1/n^{\log(n)}$, then $N = 8(\log^2(n) + 1)$.[11]

In the Public Registrar Model. Suppose that a Verifier wants to query a number of databases managed by different Provers, and engages the services of a public Registrar to help manage these interactions for them. To accomplish this, we will have the phases of the certified DP scheme proceed as follows:

- Commitment Phase: The Prover sends the Registrar the $|M| \subseteq 2^d$ monomial sum commitments so the Registrar can hold them for the Verifier(s). The Registrar can also take on performing the $n \cdot M$ parallel Σ-protocols needed to check potentially dishonest monomial commitments. Utilization of the Registrar also means this only needs to be done once overall.
- Randomness Generation Phase: The Registrar produces the random bit commitments with the Prover in parallel.
- Querying Phase: The Registrar computes $\mathbf{C}_{f(D)}$, adds on the Binomial commitment to produce $\mathbf{C}_{f(D)+\mathbf{B}_N-N/2}$, and sends the Verifier the result. The Prover sends the Verifier the opening value $f(D) + \mathbf{B}_N - N/2$ and proof $\mathbf{\Pi}_{f(D)+\mathbf{B}_N-N/2}$ as before. Then the Verifier themselves only needs to run a single **Verify** on these values.

In Table 1, the Registrar has taken on *all* of the Verifier's work except for a single **Verify**. For Pedersen commitments, this consists of two group exponentiations and one group multiplication, making the Verifier exceptionally lightweight.

[11] Note that this discussion is with respect to a per-query privacy budget of ϵ to make replicability and comparison easier, since composition will depend on the exact number and style of queries.

From a Public Registrar To Public Verifiability. It is also possible to instantiate the Registrar so that their computations can be double-checked by any Verifier. In particular, the commitment phase consists only of Σ-protocols which can be made non-interactive with Fiat-Shamir. During randomness generation, the Σ-protocol component can also be Fiat-Shamired, but the Registrar's coin-flipping-in-the-well bit must be interactively provided after the Prover's first message (as shown in Section 4.5). This single public coin for each random bit commitment could be selected according to the NIST public randomness beacon [60], so that a Verifier who was not involved in the interaction can be convinced that it was selected after the Prover's commitment (using standard security timestamping techniques). Of course, this requires that the Verifiers trust the NIST public randomness beacon, or whichever alternate source of public random bits is utilized. At query-time, a Verifier can compute the Registrar's homomorphic operations using the commitments from the first two phases.

In general, since a Registrar $\mathsf{R} = (\mathsf{R}^{reg}, \mathsf{R}^{query})$ is public-coin, a public source of trusted randomness can always be utilized to make the registration phase publicly verifiable. Since the Registrar's view maintains perfect input indistinguishability, publicly publishing the messages from registration, including the Registrar's private output $\mathbf{Info}_\theta$, doesn't leak any privacy. Then in order to process a query q, a Verifier can run $\mathsf{R}^{query}(\mathbf{Info}_\theta, q)$ for themselves.

6 Implementation of Certified DP with Pedersen Commitments

In order to ascertain the practically of this scheme and determine how our complexity analysis translates to concrete runtimes in practice, as well as directly compare to prior work, we implemented our certified Binomial mechanism to answer the following evaluation questions.

Evaluation Questions.

1. How does our implementation compare with prior work?
2. How does the runtime of each phase scale as you vary database size n and data dimension d?
3. How do these runtimes scale as you vary the *per-query* privacy budget ϵ?
4. How does query polynomial sparsity s_f impact query time?
5. How does our implementation perform in a Census-based real-world use case?

We implemented the certified Binomial mechanism in 1,479 lines of Rust, backed by curve25519-dalek, which implements Pedersen commitments over the prime-order Ristretto group [52]. We implemented Σ-protocols based on Appendix B. The Prover generates n random d-dimension database entries, which the Verifer queries by generating s_f random non-zero coefficients for some $s_f \leq 2^d$, the total number of possible monomials. We set $\delta = 1/n^{\log(n)}$, although this can be overridden. Maximum monomial degree $k \leq d$ can also be set for efficiency. We executed the parties as separate processes on a 2.7 GHz Quad-core Intel Core i7 with 16 GB RAM, communicating by TCP socket. We note that many of the protocols are easily parallelizable; as a result, we run each participant in a single thread and report single core performance. Our implementation is available at `https://github.com/jlwatson/certified-dp`.

Note that while the goal is to account for the total amount of work needed to carry out the protocol, our use of a single core here is exceedingly pessimistic in terms of runtime, and any real-world deployment should take advantage of parallelism. Since the Prover commits to monomials independently for each entry in the database, processing database elements in parallel across many cores would be straightforward and could significantly help end-to-end performance. Excepting coordination to gather the results of each core's computation and transmit them over the network, each core could commit to entries of a subset of the database as fast as our single core does in the evaluation. Further, each random bit commitment can also be performed in parallel.

Comparison to Prior Work. We compare to [15] in the case of performing single-dimension "counting" queries[12] over a large database ($n = 10^6$) of bits $d = 1$ with strict DP parameters $\epsilon = 0.095, \delta = 10^{-10}$. Table 2 compares the runtime for six different groups of operations: Verifier dishonest commitment, Prover randomness generation, Verifer randomness generation, generating and aggregating ($\oplus$) n coin flips, Prover coin flip and query aggegation, and Verifier query verification. [15]'s estimates are based on microbenchmarking individual operation counts (e.g. expected

[12][15] only supports simple summation over a database of single bits, with no predicate.

	V-Dishonest Comm. (s)	**P**-Rand. Gen. (s)	**V**-Rand. Gen. (s)	Rand. n-$\oplus$ (s)	Rand. n-$\oplus$ & Querying-$\oplus$ (ms)	Query Verify (ms)
[15]	169	53	45	33	79	189
CDP	156	4.0	5.0	0.15	7.5	0.037

Table 2: Comparative benchmark for single-dimension counting queries over a database of size $n = 10^6, d = 1$, and parameters $\epsilon = 0.095, \delta = 10^{-10}$.

number of group exponentiations) rather than implementing the full protocol as we have done, but we have matched the measured operation groups as closely as possible to provide a fair comparison. While the dishonest commitment runtime remains similar, as we also performs a Σ-protocol for each bit in the database, both Prover and Verifier randomness generation is significantly faster ($13\times$ and $9\times$, respectively). Similarly, performing and aggregating the n coin flips requires only 150 ms compared to 33 *seconds* previously on the Verifier, and coin flips and query aggregation on the Prover take 7.5 ms compared to 79 ms previously. Most importantly, query verification in our CDP prototype is orders of magnitude quicker, requiring only *37 μs*.

Database Size and Dimension. We evaluate the impact of database size n and dimension d on each phase's runtime in Fig. 4(a) and (b), respectively. Commitment time scales linearly with n and exponentially with d. In (a), holding $d = 7$ constant, the dishonest commitment phase grows from 38 s when $n = 1024$ to 619 s when $n = 16384$ (i.e., a $\sim 16\times$ increase in each parameter). Our measured honest commitment time is essentially constant (appx. 6 ms) across database sizes because the operations that scale linearly in n are in plaintext and overshadowed by the 2^d commitment operations. Through δ, the randomness generation runtime scales with $\log^2 n$, from 320 ms to 680 ms from $n = 1024$ to 16384, remaining very efficient compared to the commitment. In (b), we indeed see that both honest and dishonest commitment time scales exponentially with dimension. With $n = 1024$, increasing the dimension from $d = 7$ to 14 bits increases the runtime from 39 s to 5616 s. In practice, our prototype can reduce overall commitment time by setting a maximum degree k to support only k-wise marginals of the data. Finally, no matter n or d, the query phase remains constant, and runs exceptionally quickly (350 µs on average), compared to the expensive one-time commitment phase.

Privacy Budget. In Fig. 4(c), we evaluate system performance as our per-query privacy budget ϵ varies, measuring the runtime of each phase on a database of $n = 1024$ with $d = 7$ and $s_f = 7$. Regardless of the ϵ chosen, the honest and dishonest commitment phases remain constant, requiring about 6 ms and 39 s, respectively; similarly, queries require only 300 μs across choices of privacy parameters. The randomness generation phase scales as expected with $1/\epsilon^2$: with an $\epsilon = 10$ budget similar to those used in practice, randomness generation requires 5 ms, while a more theoretically stringent choice of $\epsilon = 1$ requires 338 ms. Finally, a very strong $\epsilon = 0.1$ incurs a 39 s runtime.

Query Polynomial Sparsity. Fig. 4(d) evaluates the impact of query sparsity on the Verifier's query time, tracking the runtime of homomorphic operations on commitments and commitment verifications on a database of size $n = 1024$, $d = 7$, with $\epsilon = 1$. As expected, homomorphic

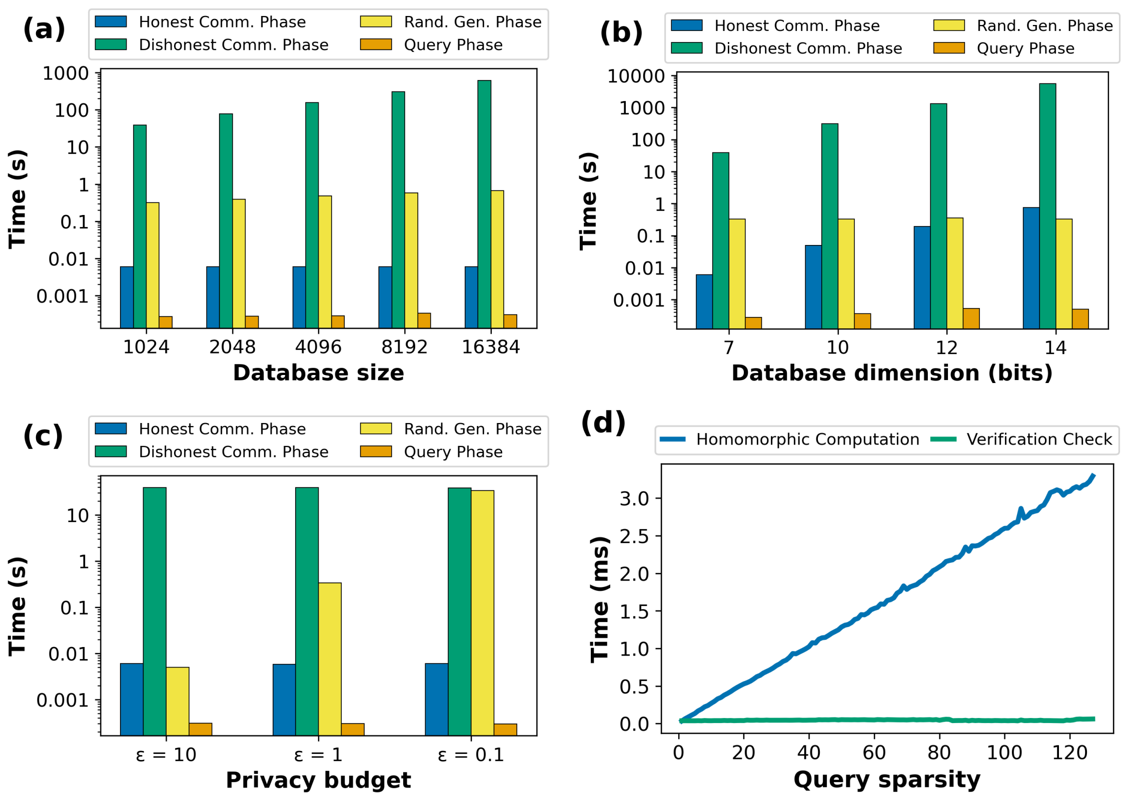


Figure 4: Experimental performance for commitment, randomness generation, and query phases, varying size n, dimension d, privacy budget ϵ, and sparsity s_f. Note that (a), (b), and (c) utilize a log scale so that the times for all phases are visible on the same axes.

computation scales linearly with query sparsity, while verification remains constant regardless of query size (or any others except the security parameter, λ). Even with maximum sparsity of $2^7 - 1 = 127$, the query phase requires only 3.3 ms, of which about 3.29 ms is homomorphic computation. The Pedersen commitment verification requires **only 61 μs for the Verifier to compute**; with a public Registrar, this is the only computation required of the Verifier, as the Registrar performs the homomorphic computation in parallel to the Prover.

Querying a Census PUMS dataset. To evaluate our prototype on a real-world use case, we downloaded a 7000-entry PUMS census dataset [23] containing anonymized age, sex, income, and education data and compressed each field into a $d = 37$ bit database entry on which to perform queries. The dimension value represents the bits of database entry information that we generate monomials over, and thus allow the client to query. For this database, there can be many fields per anonymized record (see [23]). Since we do not want to query all fields in this case, and thus generating monomials for them would be wasted work, we select 37 bits representing the fields we are interested in: 'age' (7 bits), 'sex' (1 bit), 'income' (23 bits), and 'education' (6 bits) and then run queries across those. Note however that the choice of data encoding is quite flexible and the data needn't be put into this bitwise representation that we've used for concreteness. An interesting area for future work would be considering a tailored data representation that targets a specific set of expected queries.

We set a maximum predicate degree of 6, enough to operate over the high-order income bits of each database entry, and generated a 63-sparse predicate query to count how many dataset individuals reported income in excess of \$262,144. We performed an honest commitment with the Prover, requiring 136 s, completed a randomness generation phase in $\sim$494 ms for $\epsilon = 1$, and queried in 1.9 ms. Of that, 1.6 ms consisted of homomorphic computation while verification required only 38 μs. As such, functionality (useful queries over higher-dimension data) can be balanced with efficiency (pruning the scope of supported predicates). Our Census querying experiment is implemented in a separate branch, accessible at `https://github.com/jlwatson/certified-dp/tree/census`.

Acknowledgments

This material is based upon work supported by DARPA-ORACLEs Grant contract #HR001120C0015, the Simons Collaboration on The Theory of Algorithmic Fairness, and the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 2146752. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation The authors thank Aleksandra Korolova and Raluca Ada Popa for helpful discussions, as well as several reviewers for their feedback.

References

[1] Differential privacy. Tech. rep., Apple, Inc., `https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf`

[2] United states v. meta platforms, inc., f/k/a facebook, inc. (s.d.n.y.) (2023), `https://www.justice.gov/crt/case/united-states-v-meta-platforms-inc-fka-facebook-inc-sdny`

[3] Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., Wallach, H.: A reductions approach to fair classification. In: International conference on machine learning. pp. 60–69. PMLR (2018)

[4] Agarwal, N., Suresh, A.T., Yu, F., Kumar, S., McMahan, H.B.: Cpsgd: Communication-efficient and differentially-private distributed sgd. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. p. 7575–7586. NIPS'18, Curran Associates Inc., Red Hook, NY, USA (2018)

[5] Agarwal, S., Deshpande, A.: On the power of randomization in fair classification and representation. In: Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency. pp. 1542–1551 (2022)

[6] Albarghouthi, A., Hsu, J.: Synthesizing coupling proofs of differential privacy. Proc. ACM Program. Lang. **2**(POPL), 58:1–58:30 (2018). https://doi.org/10.1145/3158146, `https://doi.org/10.1145/3158146`

[7] Albrecht, M.R., Cini, V., Lai, R.W.F., Malavolta, G., Thyagarajan, S.A.K.: Lattice-based snarks: Publicly verifiable, preprocessing, and recursively composable - (extended abstract). In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13508, pp. 102–132. Springer (2022). https://doi.org/10.1007/978-3-031-15979-4_4, `https://doi.org/10.1007/978-3-031-15979-4_4`

[8] Bamberger, K.A., Canetti, R., Goldwasser, S., Wexler, R., Zimmerman, E.J.: Verification dilemmas in law and the promise of zero-knowledge proofs. Berkeley Tech. LJ **37**, 1 (2022)

[9] Barthe, G., Chadha, R., Krogmeier, P., Sistla, A.P., Viswanathan, M.: Deciding accuracy of differential privacy schemes. Proc. ACM Program. Lang. **5**(POPL), 1–30 (2021). https://doi.org/10.1145/3434289, `https://doi.org/10.1145/3434289`

[10] Barthe, G., Gaboardi, M., Arias, E.J.G., Hsu, J., Roth, A., Strub, P.: Higher-order approximate relational refinement types for mechanism design and differential privacy. In: Rajamani, S.K., Walker, D. (eds.) Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2015, Mumbai, India, January 15-17, 2015. pp. 55–68. ACM (2015). https://doi.org/10.1145/2676726.2677000, https://doi.org/10.1145/2676726.2677000

[11] Baum, C., Orsini, E., Scholl, P., Soria-Vazquez, E.: Efficient constant-round MPC with identifiable abort and public verifiability. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12171, pp. 562–592. Springer (2020). https://doi.org/10.1007/978-3-030-56880-1_20, https://doi.org/10.1007/978-3-030-56880-1_20

[12] Benarroch, D., Campanelli, M., Fiore, D., Kim, J., Lee, J., Oh, H., Querol, A.: Proposal: commit-and-prove zero-knowledge proof systems and extensions. In: 4th ZKProof Workshop (2021)

[13] Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S.: On the dangers of stochastic parrots: Can language models be too big? In: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency. pp. 610–623 (2021)

[14] Berghel, S., Bohannon, P., Desfontaines, D., Estes, C., Haney, S., Hartman, L., Hay, M., Machanavajjhala, A., Magerlein, T., Miklau, G., Pai, A., Sexton, W., Shrestha, R.: Tumult analytics: a robust, easy-to-use, scalable, and expressive framework for differential privacy. CoRR **abs/2212.04133** (2022). https://doi.org/10.48550/ARXIV.2212.04133, https://doi.org/10.48550/arXiv.2212.04133

[15] Biswas, A., Cormode, G.: Interactive proofs for differentially private counting. In: Meng, W., Jensen, C.D., Cremers, C., Kirda, E. (eds.) Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023. pp. 1919–1933. ACM (2023). https://doi.org/10.1145/3576915.3616681, https://doi.org/10.1145/3576915.3616681

[16] Bitan, D., Canetti, R., Goldwasser, S., Wexler, R.: Using zero-knowledge to reconcile law enforcement secrecy and fair trial rights in criminal cases. Proceedings of the 2022 Symposium on Computer Science and Law (CSLAW '22), November 1–2, 2022, Washington, DC, USA (2022)

[17] Blum, A., Dwork, C., McSherry, F., Nissim, K.: Practical privacy: The sulq framework. In: Proceedings of the Twenty-Fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. p. 128–138. PODS '05, Association for Computing Machinery, New York, NY, USA (2005). https://doi.org/10.1145/1065167.1065184, https://doi.org/10.1145/1065167.1065184

[18] Blum, M.: Coin flipping by telephone a protocol for solving impossible problems. SIGACT News **15**(1), 23–27 (jan 1983). https://doi.org/10.1145/1008908.1008911, https://doi.org/10.1145/1008908.1008911

[19] Bost, R., Popa, R.A., Tu, S., Goldwasser, S.: Machine learning classification over encrypted data. Cryptology ePrint Archive (2014)

[20] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)

[21] Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y.T., Li, Y., Lundberg, S., et al.: Sparks of artificial general intelligence: Early experiments with gpt-4. arXiv preprint arXiv:2303.12712 (2023)

[22] Bureau, P.R., the U.S. Census Bureau's 2020 Census Data Products, Team, D.: Why the census bureau chose differential privacy. Tech. rep., US Census Bureau (2023), `https://www.census.gov/library/publications/2023/decennial/c2020br-03.html`

[23] Bureau, U.: Public use microdata sample (pums). Suitland, MD: The United States Census Bureau. Available online at: https://www. census. gov/programs-surveys/acs/microdata. html (accessed June 24, 2021) (2021)

[24] Canonne, C.L., Kamath, G., Steinke, T.: Discrete gaussian for differential privacy. J. Priv. Confidentiality **12**(1) (2022). https://doi.org/10.29012/JPC.784, `https://doi.org/10.29012/jpc.784`

[25] Catalano, D., Fiore, D.: Vector commitments and their applications. In: Kurosawa, K., Hanaoka, G. (eds.) Public-Key Cryptography. vol. 7778. Springer (2013), `https://doi.org/10.1007/978-3-642-36362-7_5`

[26] Catalano, D., Fiore, D., Tucker, I.: Additive-homomorphic functional commitments and applications to homomorphic signatures. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 13794, pp. 159–188. Springer (2022). https://doi.org/10.1007/978-3-031-22972-5_6, `https://doi.org/10.1007/978-3-031-22972-5_6`

[27] Chang, I., Sotiraki, K., Chen, W., Kantarcioglu, M., Popa, R.: HOLMES: Efficient distribution testing for secure collaborative learning. In: 32nd USENIX Security Symposium (USENIX Security 23). pp. 4823–4840. USENIX Association, Anaheim, CA (Aug 2023), `https://www.usenix.org/conference/usenixsecurity23/presentation/chang`

[28] Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23. pp. 409–437. Springer (2017)

[29] Cheu, A.: Differential privacy in the shuffle model: A survey of separations. arXiv preprint arXiv:2107.11839 (2021)

[30] Chiesa, A., Gur, T.: Proofs of proximity for distribution testing. In: 9th Innovations in Theoretical Computer Science Conference (ITCS 2018). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2018)

[31] Cohen, J., Rosenfeld, E., Kolter, Z.: Certified adversarial robustness via randomized smoothing. In: international conference on machine learning. pp. 1310–1320. PMLR (2019)

[32] Cramer, R., Damgard, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) Advances in Cryptology — CRYPTO '94. pp. 174–187. Springer Berlin Heidelberg, Berlin, Heidelberg (1994)

[33] Desfontaines, D.: A list of real-world uses of differential privacy. `https://desfontain.es/privacy/real-world-differential-privacy.html` (10 2021), ted is writing things (personal blog)

[34] Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)

[35] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: Proceedings of the 3rd innovations in theoretical computer science conference. pp. 214–226 (2012)

[36] Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our data, ourselves: Privacy via distributed noise generation. In: Vaudenay, S. (ed.) Advances in Cryptology - EUROCRYPT 2006. pp. 486–503. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)

[37] Dwork, C., Kohli, N., Mulligan, D.: Differential privacy in practice: Expose your epsilons! Journal of Privacy and Confidentiality **9**(2) (2019)

[38] Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3. pp. 265–284. Springer (2006)

[39] Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) Theory of Cryptography. pp. 265–284. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)

[40] Dwork, C., Rothblum, G.N., Vadhan, S.: Boosting and differential privacy. In: 2010 IEEE 51st Annual Symposium on Foundations of Computer Science. pp. 51–60. IEEE (2010)

[41] Feige, U., Shamir, A.: Witness indistinguishable and witness hiding protocols. In: Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing. p. 416–426. STOC '90, Association for Computing Machinery, New York, NY, USA (1990). https://doi.org/10.1145/100216.100272, `https://doi.org/10.1145/100216.100272`

[42] Fredrikson, M., Jha, S.: Satisfiability modulo counting: a new approach for analyzing privacy properties. In: Henzinger, T.A., Miller, D. (eds.) Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014. pp. 42:1–42:10. ACM (2014). https://doi.org/10.1145/2603088.2603097, `https://doi.org/10.1145/2603088.2603097`

[43] Gaboardi, M., Haeberlen, A., Hsu, J., Narayan, A., Pierce, B.C.: Linear dependent types for differential privacy. In: Giacobazzi, R., Cousot, R. (eds.) The 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '13, Rome, Italy - January 23 - 25, 2013. pp. 357–370. ACM (2013). https://doi.org/10.1145/2429069.2429113, https://doi.org/10.1145/2429069.2429113

[44] Gaboardi, M., Nissim, K., Purser, D.: The complexity of verifying loop-free programs as differentially private. In: Czumaj, A., Dawar, A., Merelli, E. (eds.) 47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference). LIPIcs, vol. 168, pp. 129:1–129:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). https://doi.org/10.4230/LIPICS.ICALP.2020.129, https://doi.org/10.4230/LIPIcs.ICALP.2020.129

[45] Gentry, C., Halevi, S., Vaikuntanathan, V.: A simple bgn-type cryptosystem from lwe. In: Gilbert, H. (ed.) Advances in Cryptology – EUROCRYPT 2010. pp. 506–522. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)

[46] Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., Wernsing, J.: Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In: International conference on machine learning. pp. 201–210. PMLR (2016)

[47] Gilbert, A.C., McMillan, A.: Property testing for differential privacy. In: 56th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2018, Monticello, IL, USA, October 2-5, 2018. pp. 249–258. IEEE (2018). https://doi.org/10.1109/ALLERTON.2018.8636068, https://doi.org/10.1109/ALLERTON.2018.8636068

[48] Goldwasser, S., Kim, M.P., Vaikuntanathan, V., Zamir, O.: Planting undetectable backdoors in machine learning models. In: 2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS). pp. 931–942. IEEE (2022)

[49] Goldwasser, S., Micali, S.: Probabilistic encryption & how to play mental poker keeping secret all partial information. In: Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing. p. 365–377. STOC '82, Association for Computing Machinery, New York, NY, USA (1982). https://doi.org/10.1145/800070.802212, https://doi.org/10.1145/800070.802212

[50] Goldwasser, S., Rothblum, G.N., Shafer, J., Yehudayoff, A.: Interactive proofs for verifying machine learning. In: 12th Innovations in Theoretical Computer Science Conference (ITCS 2021). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2021)

[51] Google: Differential privacy, https://github.com/google/differential-privacy

[52] Hamburg, M., de Valence, H., Lovecruft, I., Arcieri, T.: The ristretto group. https://ristretto.group/ristretto.html

[53] Hardt, M., Price, E., Srebro, N.: Equality of opportunity in supervised learning. Advances in neural information processing systems **29** (2016)

[54] Hébert-Johnson, U., Kim, M., Reingold, O., Rothblum, G.: Multicalibration: Calibration for the (computationally-identifiable) masses. In: International Conference on Machine Learning. pp. 1939–1948. PMLR (2018)

[55] Herman, T., Rothblum, G.: Doubley-efficient interactive proofs for distribution properties. In: 2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS). pp. 743–751. IEEE (2023)

[56] Herman, T., Rothblum, G.N.: Verifying the unseen: interactive proofs for label-invariant distribution properties. In: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing. pp. 1208–1219 (2022)

[57] Impagliazzo, R., Lei, R., Pitassi, T., Sorrell, J.: Reproducibility in learning. In: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing. pp. 818–831 (2022)

[58] Johnson, N.M., Near, J.P., Hellerstein, J.M., Song, D.: Chorus: a programming framework for building scalable differential privacy mechanisms. In: IEEE European Symposium on Security and Privacy, EuroS&P 2020, Genoa, Italy, September 7-11, 2020. pp. 535–551. IEEE (2020). https://doi.org/10.1109/EUROSP48549.2020.00041, https://doi.org/10.1109/EuroSP48549.2020.00041

[59] Kasiviswanathan, S.P., Lee, H.K., Nissim, K., Raskhodnikova, S., Smith, A.: What can we learn privately? In: 2008 49th Annual IEEE Symposium on Foundations of Computer Science. pp. 531–540 (2008). https://doi.org/10.1109/FOCS.2008.27

[60] Kelsey, J.: The new randomness beacon format standard: An exercise in limiting the power of a trusted third party. In: Cremers, C., Lehmann, A. (eds.) Security Standardisation Research. pp. 164–184. Springer International Publishing, Cham (2018)

[61] Lee, J.W., Kang, H., Lee, Y., Choi, W., Eom, J., Deryabin, M., Lee, E., Lee, J., Yoo, D., Kim, Y.S., et al.: Privacy-preserving machine learning with fully homomorphic encryption for deep neural network. IEEE Access **10**, 30039–30054 (2022)

[62] Libert, B., Ramanna, S.C., Yung, M.: Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions. In: Chatzigiannakis, I., Mitzenmacher, M., Rabani, Y., Sangiorgi, D. (eds.) 43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy. LIPIcs, vol. 55, pp. 30:1–30:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2016). https://doi.org/10.4230/LIPICS.ICALP.2016.30, https://doi.org/10.4230/LIPIcs.ICALP.2016.30

[63] Lipmaa, H., Pavlyk, K.: Succinct functional commitment for a large class of arithmetic circuits. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part III. Lecture Notes in Computer Science, vol. 12493, pp. 686–716. Springer (2020). https://doi.org/10.1007/978-3-030-64840-4_23, https://doi.org/10.1007/978-3-030-64840-4_23

[64] Maurer, U.: Unifying zero-knowledge proofs of knowledge. In: Preneel, B. (ed.) Progress in Cryptology – AFRICACRYPT 2009. pp. 272–286. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)

[65] Mutreja, S., Shafer, J.: Pac verification of statistical algorithms. In: Neu, G., Rosasco, L. (eds.) Proceedings of Thirty Sixth Conference on Learning Theory. Proceedings of Machine Learning Research, vol. 195, pp. 5021–5043. PMLR (12–15 Jul 2023), https://proceedings.mlr.press/v195/mutreja23a.html

[66] Narayan, A., Feldman, A., Papadimitriou, A., Haeberlen, A.: Verifiable differential privacy. In: Réveillère, L., Harris, T., Herlihy, M. (eds.) Proceedings of the Tenth European Conference on Computer Systems, EuroSys 2015, Bordeaux, France, April 21-24, 2015. pp. 28:1–28:14. ACM (2015). https://doi.org/10.1145/2741948.2741978, https://doi.org/10.1145/2741948.2741978

[67] Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) Advances in Cryptology — CRYPTO '91. pp. 129–140. Springer Berlin Heidelberg, Berlin, Heidelberg (1992)

[68] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)

[69] Raghunathan, A., Steinhardt, J., Liang, P.: Certified defenses against adversarial examples. In: International Conference on Learning Representations (2018)

[70] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-shot text-to-image generation. In: International Conference on Machine Learning. pp. 8821–8831. PMLR (2021)

[71] Rastogi, V., Nath, S.: Differentially private aggregation of distributed time-series with transformation and encryption. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data. p. 735–746. SIGMOD '10, Association for Computing Machinery, New York, NY, USA (2010). https://doi.org/10.1145/1807167.1807247, https://doi.org/10.1145/1807167.1807247

[72] Reed, J., Pierce, B.C.: Distance makes the types grow stronger: a calculus for differential privacy. In: Hudak, P., Weirich, S. (eds.) Proceeding of the 15th ACM SIGPLAN international conference on Functional programming, ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010. pp. 157–168. ACM (2010). https://doi.org/10.1145/1863543.1863568, https://doi.org/10.1145/1863543.1863568

[73] Rubinfeld, R., Vasilyan, A.: Testing distributional assumptions of learning algorithms. In: Proceedings of the 55th Annual ACM Symposium on Theory of Computing. pp. 1643–1656 (2023)

[74] Ruggles, S., Fitch, C., Magnuson, D., Schroeder, J.: Differential privacy and census data: Implications for social and economic research. In: AEA papers and proceedings. vol. 109, pp. 403–408. American Economic Association 2014 Broadway, Suite 305, Nashville, TN 37203 (2019)

[75] Shi, E., Chan, T.H., Rieffel, E.G., Chow, R., Song, D.: Privacy-preserving aggregation of time-series data. In: Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011. The Internet Society (2011), https://www.ndss-symposium.org/ndss2011/privacy-preserving-aggregation-of-time-series-data

[76] Steinke, T., Nasr, M., Jagielski, M.: Privacy auditing with one (1) training run. NeurIPS (2023)

[77] Tang, J., Korolova, A., Bai, X., Wang, X., Wang, X.: Privacy loss in apple's implementation of differential privacy on macos 10.12. arXiv:1709.02753 (2017)

[78] Thaler, J.: Proofs, arguments, and zero-knowledge. Found. Trends Priv. Secur. **4**(2-4), 117–660 (2022). https://doi.org/10.1561/3300000030, https://doi.org/10.1561/3300000030

[79] Vadhan, S.P.: The complexity of differential privacy. In: Lindell, Y. (ed.) Tutorials on the Foundations of Cryptography, pp. 347–450. Springer International Publishing (2017). https://doi.org/10.1007/978-3-319-57048-8_7, https://doi.org/10.1007/978-3-319-57048-8_7

[80] Wilson, R.J., Zhang, C.Y., Lam, W., Desfontaines, D., Simmons-Marengo, D., Gipson, B.: Differentially private SQL with bounded user contribution. Proc. Priv. Enhancing Technol. **2020**(2), 230–250 (2020). https://doi.org/10.2478/POPETS-2020-0025, https://doi.org/10.2478/popets-2020-0025

[81] Wu, J., Chen, Y., Liu, Y.: Metric-fair classifier derandomization. In: International Conference on Machine Learning. pp. 23999–24016. PMLR (2022)

[82] Zhang, D., Kifer, D.: Lightdp: towards automating differential privacy proofs. In: Castagna, G., Gordon, A.D. (eds.) Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017. pp. 888–901. ACM (2017). https://doi.org/10.1145/3009837.3009884, https://doi.org/10.1145/3009837.3009884

[83] Zou, A., Wang, Z., Kolter, J.Z., Fredrikson, M.: Universal and transferable adversarial attacks on aligned language models. arXiv preprint arXiv:2307.15043 (2023)

A The Dishonest Commitment Phase Case

A.1 Certified Probabilistic Mechanisms Definition with Dishonest Commitment Phase

Before we get to the main definition, it will simplify the presentation to define a notion of "a valid commitment to a known database." This captures the idea that after the commitment phase, the Prover should know a well-formed input θ that $\mathbf{C}_\theta$ is a commitment to, so that we can define what the output from the querying phase should be, namely $\mathsf{M}_q(\theta)$.

Definition A.1 (Valid Commitment to Known Database)**.** For a mechanism M_Q and input $\theta \in \Theta$, the pair $(\mathbf{C}, \mathbf{View})$ is a valid M_Q-commitment to known input θ if there exists a PPT algorithm **RevealOpening** that, given the Prover's view **View** and rewindable black-box access to the Prover, returns θ and $\mathbf{\Pi}$ such that the following holds.

For all $q \in Q$, let $\mathbf{Output}_q \leftarrow (\mathsf{P}^{open}(\mathbf{\Pi}), \mathsf{V}^{open})(\mathbf{C}, q)$ be the output of the querying phase *according to the honest Prover and Verifier.* Then the random variable $\mathbf{Output}_q$ is distributed according to the mechanism $\mathbf{Output}_q \overset{d}{=} \mathsf{M}_q(\theta)$.

Note that for the honest Prover, RevealOpening is trivial: the honest Prover's transcript includes θ and $\mathbf{\Pi}$, so they can simply output them. We will describe this by saying that $(\mathbf{C}, \mathbf{\Pi})$ is a valid commitment, where **RevealOpening** is now implicitly the identity function. So, this condition is only a non-trivial constraint when discussing adversarial Provers. Now we are ready to present the formal definition.

Definition A.2 (Certified Probabilistic Mechanism)**.** Given mechanisms $\mathsf{M}_q : \Theta \to Y$ for query class Q, a *certified probabilistic mechanism for* M_Q consists of **Setup**, an honest Prover $\mathsf{P} = (\mathsf{P}^{com}, \mathsf{P}^{open})$, and an honest Verifier $\mathsf{V} = (\mathsf{V}^{com}, \mathsf{V}^{open})$ with the following properties for public parameters $pp \leftarrow \mathsf{Setup}(1^\lambda)$,[13] input $\theta \in \Theta$, and query $q \in Q$:

- *Correctness*: Let $(\mathbf{C}_\theta, (\mathbf{\Pi}_\theta)_{\mathsf{P}^{com}}) \leftarrow (\mathsf{P}^{com}(\theta), \mathsf{V}^{com})$. Then $(\mathbf{C}_\theta, \mathbf{\Pi}_\theta)$ is a valid M_Q-commitment to θ.
- *Commitment Phase Soundness*:
  - *Cheating-Verifier Commitment Soundness*: For any adversary $\widetilde{\mathsf{V}}^{com}$, let $(\mathbf{C}_\theta, (\mathbf{\Pi}_\theta)_{\mathsf{P}^{com}}) \leftarrow (\mathsf{P}^{com}(\theta), \widetilde{\mathsf{V}}^{com})$. Then $\mathbf{C}_\theta = \perp$ or $(\mathbf{C}_\theta, \mathbf{\Pi}_\theta)$ is a valid M_Q-commitment to θ.
  - *Cheating-Prover Commitment Soundness*: For any PPT adversary $\widetilde{\mathsf{P}}^{com}$, let $(\mathbf{C}, \mathbf{View}_{\widetilde{\mathsf{P}}^{com}}) \leftarrow (\widetilde{\mathsf{P}}^{com}, \mathsf{V}^{com})$. Then either $\Pr[\mathbf{C} = \perp] = 1 - \text{negl}(\lambda)$ or $(\mathbf{C}_\theta, \mathbf{View}_{\widetilde{\mathsf{P}}^{com}})$ is a valid M_Q-commitment to some $D \in \Theta$. Denote $\widetilde{\mathsf{P}}^{com}$ such that the above openability property holds for database D as $\widetilde{\mathsf{P}}^{com}(\theta)$.[14]

[13]Note that we assume that all functions have implicit access to the public parameters pp and all probabilistic statements are also over the randomness used by **Setup**.

[14]If $\mathbf{C} = \perp$ the protocol terminates and does not progress to the querying phase, so for the querying phase we will only consider cheating Provers who have committed to a well-formed database in this sense.

- *Querying Phase Soundness*:

  - *Cheating-Verifier Querying Soundness*: For any adversary $\widetilde{\mathsf{V}} = (\widetilde{\mathsf{V}}^{com}, \widetilde{\mathsf{V}}^{open})$, let $(\mathbf{C}_\theta, \mathbf{\Pi}_\theta, \mathbf{View}_{\widetilde{\mathsf{V}}^{com}(\theta)}) \leftarrow (\mathsf{P}^{com}(\theta), \widetilde{\mathsf{V}}^{com})$ and $(\mathbf{Output}, \mathbf{Q}, \mathbf{View}_{\widetilde{\mathsf{V}}(\theta)}) \leftarrow (\mathsf{P}^{open}(\mathbf{\Pi}_\theta), \widetilde{\mathsf{V}}^{open}(\mathbf{View}_{\widetilde{\mathsf{V}}^{com}(\theta)}))(\mathbf{C}_\theta, q)$. Then for any $\theta, \theta' \in \Theta$ and $y = Y \cup \{\perp\}$,

  $$\mathbf{View}_{\widetilde{\mathsf{V}}(\theta)}|_{\mathbf{Q}=y} \overset{d}{=} \mathbf{View}_{\widetilde{\mathsf{V}}(\theta')}|_{\mathbf{Q}=y}.$$

  Further, let $\mathbf{Q}_{\neq\perp}$ denote $\mathbf{Q}$ conditioned on it not equalling $\perp$. Then

  $$\mathbf{Q}_{\neq\perp} \overset{d}{=} \mathsf{M}_q(\theta).$$

  - *Cheating-Prover Querying Soundness*: For any PPT adversary $\widetilde{\mathsf{P}} = (\widetilde{\mathsf{P}}^{com}(\theta), \widetilde{\mathsf{P}}^{open})$, let $(\mathbf{C}_\theta, \mathbf{View}_{\widetilde{\mathsf{P}}^{com}}) \leftarrow (\widetilde{\mathsf{P}}^{com}(\theta), \mathsf{V}^{com})$ and $\mathbf{Output} \leftarrow (\widetilde{\mathsf{P}}^{open}(\mathbf{View}_{\widetilde{\mathsf{P}}^{com}(\theta)}), \mathsf{V}^{open})(\mathbf{C}_\theta, q)$. Then

  $$\mathbf{Output} \overset{d}{\approx}_{(\Pr[\mathbf{Output}=\perp]+\mathrm{negl}(\lambda))\text{-TV}} \mathsf{M}_q(\theta).$$

Cheating-Verifier commitment soundness ensures that even with a cheating Verifier, an honest Prover only produces valid commitments. Cheat-Prover commitment soundness ensures that after the commitment phase, either the Prover has committed to a well-formed database or the honest Verifier catches them with high probability.

A.2 Certified Differential Privacy Construction with Dishonest Commitment Phase

To remove the honest commitment phase assumption from our constructions, we can use standard WI-PoK Σ-protocols for additively homomorphic commitments schemes from Appendix B to ensure that $\mathbf{C}_D$ is well-formed.

Specifically for F_{Sum}, to ensure that each $x_i \in \{0, 1\}$, we can execute the Σ-protocol for

$$\text{WI-PoK}\left\{r \mid \mathsf{Commit}(0, r) = \mathbf{C}_{x_i} \ \vee\ \mathsf{Commit}(1, r) = \mathbf{C}_{x_i}\right\}.$$

These can be done in parallel for each x_i, which constitutes the Σ-protocol for

$$\text{WI-PoK}\left\{r_1, \ldots, r_n \;\middle|\; \bigwedge_{i=1}^{n} (\mathsf{Commit}(0, r_i) = \mathbf{C}_{x_i} \ \vee\ \mathsf{Commit}(1, r_i) = \mathbf{C}_{x_i})\right\}.$$

For F_{Count}, we can start by doing the same for each $x_{i,j}$ and then build these up into all of the monomials $m_{i,S}$ for $i \in [n]$, $S \subseteq [d]$. This can be done with the Σ-protocol for OR, given in Appendix B, as well as the following one for multiplication that is noted in [64]:

$$\begin{aligned}\text{WI-PoK}\{x, x_1, x_2, r, r_1, r_2 \mid\ & \mathsf{Commit}(x, r) = c \wedge \mathsf{Commit}(x_1, r_1) = c_1 \\ & \wedge \mathsf{Commit}(x_2, r_2) = c_2 \wedge x = x_1 \cdot x_2\}.\end{aligned}$$

We specifically implement Section 12.3.2, Protocol 10 of [78]. Again, we can simply AND over all of the statements we need to build a Σ-protocol for

$$\text{WI-PoK}\left\{ m_{i,S}, r_{i,S} \;\middle|\; \bigwedge_{i=1}^{n} \left(\bigwedge_{j=1}^{d} \mathsf{Commit}(0, r_{i,j}) = \mathbf{C}_{x_{i,j}} \;\vee\; \mathsf{Commit}(1, r_{i,j}) = \mathbf{C}_{x_{i,j}} \right) \wedge \right.$$
$$\left. \left(\bigwedge_{S \subseteq [d]} \left(\mathsf{Commit}(x, r_{i,S}) = \mathbf{C}_{m_{i,S}} \wedge x = m_{i,S} \right) \right) \right\}.$$

Describing this systematically, for each $i \in [n]$ we can start by checking $x_{i,1} \in \{0,1\}$ and then proceed inductively to build up the monomials. Let M be the set of monomials that have been checked so far. Then going forward, when checking monomials utilizing $x_{i,j}$, we add the following checks to the AND of our PoK:

- first,
$$\text{WI-PoK}\left\{ r_{i,j} \mid \mathsf{Commit}(0, r_{i,j}) = \mathbf{C}_{x_{i,j}} \;\vee\; \mathsf{Commit}(1, r_{i,j}) = \mathbf{C}_{x_{i,j}} \right\},$$

- and then for each $m_{i,S} \in M$,
$$\text{WI-PoK}\{ m_{i,S\cup\{j\}}, r_{i,S\cup\{j\}} \mid \mathsf{Commit}(m_{i,S\cup\{j\}}, r_{i,S\cup\{j\}}) = \mathbf{C}_{m_{i,S\cup\{j\}}}$$
$$\wedge\, m_{i,S\cup\{j\}} = m_{i,S} \cdot x_{i,j} \}.$$

Thus the PoK statement can be written with $n \cdot 2^d$ ANDs between base Σ-protocols (which don't scale with n or d), so the number of elements sent during each round of the Σ-protocol is $O(n \cdot 2^d)$. Note that these kinds of Σ-protocols can also be made non-interactive in the Random Oracle Model via standard Fiat-Shamir.

Finally, the Verifier can use additive homomorphism to compute each $\mathbf{C}_{m_S} = \bigoplus_{i \in [n]} \mathbf{C}_{m_{i,S}}$, which also scales with $O(n \cdot 2^d)$, and deterministically check if these commitment values give rise to the correct Merkle root (or other kind of vector commitment) that has been used to succinctly summarize them.

Remark A.1 (Retroactive Commitment Verification)**.** Note that a Verifier can ask the Prover to execute these PoKs at any point in order to order to establish that the original commitment was to a well-formed database. In particular, this could be done by the Registrar or some sort of Auditor with more computational resources than the typical Verifiers, or by particularly vigilant Verifiers, either during the commitment phase as described in our formal definition or after the fact if suspicious behavior has been detected during the querying phase.

Remark A.2 (Commitment Soundness Without Verification)**.** Without this commitment-checking process, the guarantee given by our construction is simply that on query f, the output will be distributed as $y_f + \mathbf{B}_N$ for some fixed $y_f \in Y$. In particular, there may or may not exist some well-formed database D such that $f(D) = y_f$ for each $f \in F$. Note that this is the typical binding guarantee for functional commitment schemes.

B Σ-Protocols and XOR for Additively Homomorphic Commitments

To present the PoK and homomorphic XOR constructions utilized in *Construction* 4.1, it will behoove us to slightly change our commitment scheme notation for this section. We will assume that commitment scheme being used has additive homomorphism.

Notation B.1. The commitment function can be expressed deterministically with the Prover's random string as input, namely as $\mathsf{Commit}(x, r)$ for $x \in X$ and r a random string, say $r \in R$. Let the set of possible commitments output be C. For an additively homomorphic scheme, X is a group with operation $\cdot$, R is a group with operation $\star$, and C is a group with operation $\otimes$ such that $\mathsf{Commit}(x, r)$ is a homomorphism from $X \times R \to C$. Going forward, we will slightly overload notation and use $\cdot$ to denote the group operations of X, R, and $X \times R$. When the operation is clear from the context of the group, repeatedly applying an operation will be denoted by exponentiation of the group element by the appropriate integer. Now we can let r act as the proof, so that $\mathsf{Verify}(c, r, x)$ returns x if $\mathsf{Commit}(x, r) = c$ and rejects otherwise.

Construction B.1 ([32])**.** The following fulfills Theorem 2.4 by providing Σ-protocol for

$$\text{PoK}\left\{r \mid \mathsf{Commit}(x_0, r) = c \ \vee \ \mathsf{Commit}(x_1, r) = c\right\} :$$

1. DE selects $r_{\mathbf{B}_{\mathsf{DE}}} \sim \mathbb{Z}/q\mathbb{Z}$ u.a.r. and computes $c_{\mathbf{B}_{\mathsf{DE}}} = \mathsf{Commit}(x_{\mathbf{B}_{\mathsf{DE}}}, r_{\mathbf{B}_{\mathsf{DE}}})$. DE also selects $e_{1-\mathbf{B}_{\mathsf{DE}}}, z_{1-\mathbf{B}_{\mathsf{DE}}} \sim \mathbb{Z}/q\mathbb{Z}$ u.a.e. and computes $c_{1-\mathbf{B}_{\mathsf{DE}}} = \mathsf{Commit}((1 - \mathbf{B}_{\mathsf{DE}}) \cdot (1 + e_{1-\mathbf{B}_{\mathsf{DE}}}), z_{1-\mathbf{B}_{\mathsf{DE}}}) \ominus (e_{1-\mathbf{B}_{\mathsf{DE}}} \otimes \mathbf{C}_{\mathsf{DE}})$. DE sends c_0 and c_1.
2. PL selects challenge $e \sim \mathbb{Z}/q\mathbb{Z}$ u.a.e. sends it.
3. DE computes a new challenge $e_{\mathbf{B}_{\mathsf{DE}}} = e - e_{1-\mathbf{B}_{\mathsf{DE}}}$. Next DE computes $z_{\mathbf{B}_{\mathsf{DE}}} = r_{\mathbf{B}_{\mathsf{DE}}} \cdot r^{e_{\mathbf{B}_{\mathsf{DE}}}}$. DE sends e_0, e_1, z_0, z_1.
4. PL checks whether
   - $e = e_0 + e_1$,
   - $\mathsf{Commit}(x_0 \cdot (1 + e_0), z_0) = c_0 \oplus (e_0 \otimes \mathbf{C}_{\mathsf{DE}})$, and
   - $\mathsf{Commit}(x_1 \cdot (1 + e_1), z_1) = c_1 \oplus (e_1 \otimes \mathbf{C}_{\mathsf{DE}})$,

   accepting only if all three checks pass.

Construction B.2. The following provides a construction for homomorphically XORing a bit commitment (c_1, π_1) with a plaintext bit b_2:

$$(c_{1+2}, \pi_{1+2}) = \begin{cases} (c_1, \pi_1) & b_2 = 0 \\ (\mathsf{Commit}(1, 0) \oplus (-1 \otimes c_1), -1 \otimes \pi_1) & b_2 = 1 \end{cases}$$

Depending on the value of b_2, this construction either lets the committed bit b_1 stay the same or flips it to $1 - b_1$. In the second case, the commitment to the public value of 1 needs to be reproducible for the Player so we deterministically commit with fixed randomness equal to 0.