PRIME: Differentially Private Distributed Mean Estimation with Malicious Security

Laasya Bangalore *
Georgetown University

Albert Cheu †
Georgetown University

Muthuramakrishnan Venkitasubramaniam
Georgetown University

Abstract

Distributed mean estimation (DME) is a fundamental and important task as it serves as a subroutine in convex optimization, aggregate statistics, and, more generally, federated learning. The inputs for distributed mean estimation (DME) are provided by clients (such as mobile devices), and these inputs often contain sensitive information. Thus, protecting privacy and mitigating the influence of malicious adversaries are critical concerns in DME. A surge of recent works has focused on building multiparty computation (MPC) based protocols tailored for the task of secure aggregation. However, MPC fails to directly address these two issues: (i) the potential manipulation of input by adversaries, and (ii) the leakage of information from the underlying function. This paper presents a novel approach that addresses both these issues. We propose a secure aggregation protocol with a robustness guarantee, effectively protecting the system from "faulty" inputs introduced by malicious clients. Our protocol further ensures differential privacy, so that the underlying function will not leak significant information about individuals. Notably, this work represents the first comprehensive effort to combine robustness and differential privacy guarantees in the context of DME. In particular, we capture the security of the protocol via a notion of "usefulness" combined with differential privacy inspired by the work of Mironov et al. (CRYPTO 2009) and formally analyze this security guarantee for our protocol.

*This author is currently at SandboxAQ.
†This author is currently at Google Research.

1 Introduction

In *distributed mean estimation* (DME), numerical data are spread over a large number of clients, and the objective is to compute the mean of those data. This fundamental task serves as a core subroutine for implementing gradient descent in convex optimizations, aggregate statistics, and, more generally, federated learning. In most cases, individual data is sensitive and we need to perform aggregation in a privacy-preserving manner. Distributed secure aggregation has become an active area of research with a surge of works focusing on secure aggregation with varying security/privacy guarantees [26, 14, 11, 10, 63]. Secure multiparty computation (MPC) serves as the right tool for DME as it enables distributed computation of arbitrary tasks while simultaneously guaranteeing *privacy* where nothing beyond the output of the computation is revealed and *correctness* where the output is correct according to the specified function. Furthermore, MPC provides these guarantees even in the presence of an adversary that can control a subset of the parties and launch a coordinated attack on the protocol. While MPC shows *how to compute* in a distributed environment with the best possible security, there are two issues MPC fails to address: (1) it does not automatically prevent adversaries from setting the inputs of corrupted parties arbitrarily thereby affecting the accuracy of the computation and (2) it does not quantify *what is leaked in the computation* as in many cases the underlying function (e.g. sum of the inputs) itself can leak information of parties' inputs.

Toward mitigating the first of these concerns, Corrigan-Gibbs and Boneh [26] designed the Prio system, a robust secure aggregation protocol. Prio is widely popular and has been incorporated in the works of Apple, Google, Internet Services Research Group (ISRG), and Mozilla. For example, Mozilla uses a modified version of Prio to collect web usage statistics privately. Apple and Google use Prio for their exposure notifications express (ENX) system. In the Prio architecture, a set of clients holding private inputs delegates the task of aggregation to a set of servers. The Prio protocol achieves privacy of an honest client's input even in the presence of a semi-honest (passive) adversary that corrupts an arbitrary subset of the servers. A key feature of their work is the *robustness* guarantee that protects the system from "faulty" inputs. More precisely, given some polynomial-time computable predicate P their system is able to discard bad inputs, i.e., those that do not satisfy the predicate P via a *input certification* mechanism. Another attractive feature of their work is that the clients only need to send a single (i.e. non-interactive) message to all the servers. This feature allows the clients to participate over weak networks. The main drawbacks of their protocol are that they tolerate only semi-honest or passive corruption of the servers and incur large communication cost between the clients and servers.[1] A follow-up work by Talwar [63], still in the semi-honest model, improves the efficiency of Prio under the same threat model and architecture by making two relaxations. First, the robustness guarantee is relaxed to approximate robustness where invalid inputs could be accepted with small (yet, non-negligible) probability. Second, the security guarantee allows differentially private leakage of the honest clients' inputs in addition to the output of the computation.[2]

Another line of work, initiated by Bonawitz et al. [14] considers secure aggregation in the star topology where the central node is connected with all the clients and learns the result of the secure aggregation. The main feature of their construction, orthogonal to the Prio line of works, is that the protocol execution goes to completion even when a subset (up to a threshold) of nodes drop out or behave maliciously. This is referred to as *guaranteed output delivery* in the MPC literature. A series of works [11, 10, 7, 48] have refined this approach to additionally achieve robustness. One drawback that persists in this line of works is that they achieve security only against semi-honest corruption of the center node. A recent work shows how to achieve security against malicious corruption of the center node but this comes at the price of trading the guaranteed output feature for *security with abort* [10].

Finally, no prior work addresses information leakage from the underlying function. In particular, none of the previous works incorporate the stronger guarantee of *differentially privacy* (DP) [30] with a formal analysis on the impacts on accuracy (See Section B). In particular, no previous work shows how to use robust aggregation in the context of differential privacy. This intersection is non-trivial to study because we must

[1] The communication between every client and each server is proportional to "circuit" size of the predicate P.

[2] In contrast, standard MPC security guarantees nothing beyond the output of the computation is revealed.

account for *manipulation attacks* [23]. The output of any DP protocol has error due to noise and malicious clients can always pretend to have input satisfying predicate P to add to that error—for example, increase bias in a mean estimate by faking large-norm input—but there exist attacks that introduce even more error in the outcome. We give a more rigorous definition later in our work. In summary, we are interested in the following question:

> *Is it possible to achieve the best of all worlds: perform differentially private distributed mean estimation while satisfying guaranteed output delivery and mitigating the threat of manipulation attacks, with a malicious adversary?*

In slightly more detail, our goal is to design a secure aggregation protocol in the Prio architecture that meets the following desiderata:

(a) Privacy: A system is said to be private if the view of the adversary that launches a coordinated attack is differentially private [30].

(b) Guaranteed output delivery: We require that the execution goes to completion with the honest parties learning the output of the computation no matter how the adversary tries to sabotage the execution.

(c) Guaranteed input inclusion: The inputs of all honest parties should be included in the computation even if the adversary tries to actively attack the system.[3]

(d) Input Certification (a.k.a. robustness w.r.t. predicate P): In a robust secure aggregation system, corrupted clients should be prevented from giving "artificial" inputs. If the underlying domain D can be captured via a predicate $P : \mathbb{F}^m \to \{0,1\}$ which outputs 1 on all inputs $x \in D$ and 0 otherwise, then a simple form of robustness allows aggregation of inputs if and only if the predicate on its input returns 1 [20, 15, 53].

(e) Enable lightweight clients on unreliable networks: In order to accommodate lightweight clients that transmit information from unreliable networks, the system should impose minimal cryptographic operations on the client side and avoid multiple rounds of interaction with the clients.

1.1 Our Contributions

In this work, we design a concretely efficient protocol in the Prio architecture that satisfies our desiderata. We proceed in two modular steps. First, we build a client-server protocol for robust[4] secure addition of vectors that achieves full security in the presence of a malicious adversary that corrupts an arbitrary number of clients and $1/3^{rd}$ of the servers[5].We remark that ours is the first concretely efficient protocol that guarantees security against a malicious adversary. Roughly speaking, our overall communication is proportional to sum of the best protocol for *verifiable secret sharing* and (succinct) zero-knowledge arguments for predicate P. As a by-product, we also obtain a concretely efficient verifiable relation sharing scheme in the random oracle model [4]. More formally, we obtain the following theorem.

Theorem 1.1 (Input Certified Secure Aggregation)**.** *Let $\mathsf{n}_s, \mathsf{t}_\mathsf{s} \in \mathbb{N}$ such that $\mathsf{t}_\mathsf{s} < \mathsf{n}_s/3$ and $P : \mathbb{F}^d \to \{0,1\}$ be an arbitrary predicate. Let $\mathcal{F}_{\mathsf{Agg}}$ be the ideal functionality given in Figure 4. The protocol Π_{Agg}, as outlined in Figure 10, securely realizes $\mathcal{F}_{\mathsf{Agg}}$ among n_c clients each holding input vectors of length d with elements in some finite field $\mathbb{F}$, n_s servers, and an output party $\mathcal{O}$, which is secure against a static, malicious rushing adversary that can maliciously corrupt an arbitrary number of clients, up to t_s servers and the output party and ensures guaranteed output delivery where κ is the security parameter. Additionally, a client is required to engage in only a single round of communication.*

[3] Typically, guaranteed output delivery implies guaranteed input inclusion. However, in scenarios where the input parties can join in a permissionless way and the adversary controls who can join (as is the case in the single server setting described later), guaranteed input inclusion does not hold.

[4] For some polynomial-time computable predicate P, robust secure addition only sums the inputs from users for which the predicate returns true. In other words, P helps to weed out "bad" inputs.

[5] It is well known that full security cannot be achieved when there is no honest majority [25].

Moreover, the total communication complexity between a client and each server is $O(d + \mathbb{d} \log |P| + \log^2(\mathsf{n}_s \cdot d))$ field elements where $|P|$ and $\mathbb{d}$ are the size and depth (respectively) of the circuit associated with the predicate P. The total amortized communication among the servers is $O(\mathsf{n}_\mathsf{c} \cdot d \cdot \mathsf{n}_s^3)$ field elements in the offline phase and $O(\mathsf{n}_\mathsf{c} \cdot d \cdot \mathsf{n}_s^2)$ field elements in the online phase for a sufficiently large d.

Second, we design a differentially private mechanism for DME that uses the secure aggregation protocol from the first step as a black box. The mechanism is specified through algorithms PRE and POST along with a predicate P for the secure aggregation protocol, denoted Π_{Agg}. Clients first pre-process their input using PRE, then feed it as input to Π_{Agg}. The predicate P filters out inputs with an overly large norm. The output party processes the aggregation using POST to obtain an estimate of the mean. Slightly more formally, we obtain the following theorem:

Theorem 1.2 (Upper bound for DP DME (Informal))**.** *Let (ε, δ) be target privacy parameters. Assume client data X_i ($i \in [n]$) belongs to the Euclidean unit ball $\mathcal{B}^d$. There is a predicate P and a way to pre-process X_i into Y_i such that the composition of Π_{Agg} and pre-processing is (ε, δ)-differentially private. Moreover, the output party can post-process the aggregated value to obtain an unbiased estimate of the mean $\frac{1}{n}\sum X_i$, with variance $O(\frac{d}{\varepsilon^2 n^2} \log \frac{1}{\delta})$, asymptotically the same as the classic Gaussian mechanism.*

If $t = O(n)$ clients are malicious, then we achieve $(O(\varepsilon), O(\delta))$-differential privacy and the expected squared error $\tilde{O}(\frac{t^2}{n^2} \cdot \frac{d}{\varepsilon^2 n})$.

As our final contribution we argue that a natural class of PRE, POST algorithms cannot have asymptotically better accuracy. More formally, we prove the following theorem.

Theorem 1.3 (A Lower bound for Wrapped DP DME (informal))**.** *For any pair of pre- and post-processing algorithms that preserves $(\varepsilon, \delta = o(1/n))$-differential privacy where additionally the post-processing algorithm is affine, the resulting system* either *produces biased estimates when there are no malicious clients* or *there is an explicit attack by t malicious clients that results in $\frac{t^2}{n^2} \cdot \frac{d}{\varepsilon^2 n}$ expected squared error.*

On Minimality of our Assumptions. Although robustness (i.e., input certification) can be achieved without any assumptions through distributed zero-knowledge proofs, for the sake of efficiency (and enabling lightweight clients), we utilize succinct non-interactive arguments that rely solely on symmetric-key primitives. To achieve guaranteed output delivery[6], we require an honest majority among the servers. While $n \geq 2t + 1$ suffices, we adopt a stronger condition of $n \geq 3t + 1$ to enhance efficiency. Specifically, our protocol relies on verifiable secret sharing[7], which in the $n \geq 2t + 1$ setting often requires quadratic communication complexity (in n) due to the reliance on bivariate polynomials [45, 54, 5] or involves a trusted setup that is computationally expensive [44]. Such inefficiencies are undesirable in our context, given the unreliability of network conditions. Lastly, our final protocol achieves only computational differential privacy as opposed to information-theoretic differential privacy since we rely on zkSNARKs for efficiency that exist only based on computational assumptions.

1.2 Our Techniques

In this section, we give a brief overview of our techniques. On a high level, we rely on differential privacy to determine *what to compute* and secure multiparty computation to design *how to compute*.

Prior to outlining the techniques, we identify the types of attacks these techniques aim to counter. Essentially, these can be classified into two main groups: attacks that compromise the integrity of the computation and attacks that violate the privacy of the input.

[6]Guaranteed output delivery inherently takes into account security against malicious adversaries, since it trivially holds when only semi-honest adversaries are present.

[7]More precisely, we utilize verifiable relation sharing.

Privacy attacks can be further classified into two categories. The first captures the ability of the adversary to learn "more" than the output of the pre-specified functionality. The MPC guarantees of our secure aggregation protocol defends against this attack. The second privacy attack captures the ability of the adversary to learn "more" by merely chosing the inputs of the controlled parties and observing the output. Differential privacy is a tool to address this attack.

Integrity attacks, on the other hand, come in three varieties. The first is the denial of service attack: the adversary prevents the protocol from producing output. The second is against simulation correctness: the adversary tries to force the protocol to produce an output other than $f(X)$ for pre-specified function f and inputs held by the honest parties. Our secure aggregation protocol is additionally robust and hence can defend against these two attacks, as it guarantees output even in the presence of a malicious adversary. The third integrity attack is the manipulation attack: the adversary tries to force the protocol to produce something more than α-far from $g(X)$, for positive real α and g (in a metric space).

A straightforward approach to achieve our goals would be to rely on an MPC protocol (with guaranteed output delivery) to securely realize a differentially private mechanism for secure aggregation. Robustness can be incorporated into the functionality, where only inputs that satisfy the predicate will be included in the computation. Although the resulting protocol will provide simulation-based security, a separate analysis is needed on the accuracy of the functionality in the presence of a malicious adversary. Furthermore, getting a concretely efficient protocol following this approach is hard because such a functionality is randomized and the servers would have to generate the noise needed for DP mechanisms in a distributed way. The protocol of Dwork et al. [29] gives an elegant mechanism for distributed generation of Gaussian noise, however, incorporating robustness would still be a challenge. In this work, we take a different path. Instead of aiming for a simulation-based security, we will consider a meaningful relaxation of the security that will be adequate and lead to a more concretely efficient solution. We believe that this definition could be of independent interest for combining differential privacy and MPC guarantees against malicious adversaries.

We capture our security requirements by defining *usefulness* and *privacy* in the presence of malicious adversaries inspired by the work of Mironov et al. [50] who defined similar notions for semi-honest adversaries. Capturing usefulness in the malicious setting is hard since accuracy can be identified only w.r.t. honest party's inputs. Our approach is to avoid identifying honest parties and target the goal of the underlying task, namely, DME and we define a protocol useful if it can compute the sample mean assuming the honest parties' inputs come from a predefined distribution. Privacy, on the other hand, extends the indistinguishability-based computational differential privacy notion from [50] and requires that the view of the malicious adversary is (ϵ, δ) differentially private.

We will be able to modularly present our techniques for secure aggregation via MPC and DP. In slightly more detail, our final protocol will be able to combine a robust protocol for secure summation with a DP mechanism in essentially a black-box way that we explain next.

1.2.1 Secure Aggregation.

We present our input-certified secure aggregation protocol (in short, certified aggregation) where n_s servers $\mathcal{S}$ securely aggregate the inputs of n_c clients $\mathcal{U} = \{u_1, \ldots, u_{\mathsf{n}_\mathsf{c}}\}$ that satisfy a predicate $P(\cdot)$. More precisely, each client $u_i \in \mathcal{U}$ has a private input vector $X_i \in \mathbb{F}^d$. At a high level, the goal of the protocol is to compute $\sum_{u_i \in \mathcal{U}} X_i \cdot P(X_i)$ in a secure manner, i.e. an adversary maliciously corrupting up to a third of the servers and any number of clients does not learn anything beyond the output. Our protocol follows the standard template of MPC protocols to securely compute the aggregate where the clients secretly share their inputs among the servers, and the servers perform the computation. To protect against a malicious adversary, we need to ensure that the shares are well formed and the secret input of the client satisfies a predicate $P(\cdot)$. Traditionally, verifiable secret sharing (VSS) schemes have been employed to guarantee that the client secret-shares the input correctly.

Our starting point is the work of Zhang et al. [68] who provide a concretely efficient VSS scheme from a polynomial commitment scheme. Here, the dealer commits to a polynomial and proves that the polynomial

is evaluated correctly at n different evaluation points for each verifier. While their VSS scheme boasts desirable properties like a lack of trusted setup and optimal prover time[8], it poses two significant challenges in our context.

First, in our setting, we point out that for robustness, the clients additionally have to prove that their secret input satisfies a certain predicate. This primitive is referred to as a verifiable relation sharing scheme (implicitly) introduced in the work of Gennaro et al. [37] and formally by Applebaum et al. [4]. Second, in typical VSS (and existing VRS schemes [4]), the dealer needs to participate in more than one round. In our setting, to address client dropouts, we impose the requirement that the clients send a single message (to each of the servers).

To address the first issue, we need each client (acting as a prover) to share a vector (of secrets) to the servers while proving to each of the servers in zero-knowledge that the shared data satisfies a pre-specified predicate. As mentioned above, we will rely on a Verifiable Relation Sharing (VRS) scheme. We extend the construction of Zhang et al. [68] to build the desired VRS scheme. In more detail, we first formalize their generalized polynomial commitment scheme[9] as a distributed commit and prove (dCP) functionality and then instantiate it using a variant of the construction in [68]. At a high level, the dCP functionality is parameterized by a tuple of relations $(\mathcal{R}_1, \ldots, \mathcal{R}_n)$ that allows a prover to commit to a value, say w, to a set of verifiers in a commit phase and later in a prove phase prove to each verifier $\mathcal{V}_i$ that $(x_i, w) \in \mathcal{R}_i$[10]. A central construction in this work is essentially a modular design of a VRS scheme built from a VSS scheme and a dCP protocol.

Next, addressing the second challenge of reducing the dealer's participation to a single round, we propose a new approach inspired by protocols in the proactive secret-sharing literature [39, 16, 64] At a high level, in our VRS scheme, the dealer participates in one round to secret-share its input and prove it satisfies an underlying predicate via the dCP functionality. Subsequently, the verifiers interact among themselves to verify if the dealer performed the sharing correctly, without the dealer's involvement. If any of the verifiers detect missing shares or fail to verify the proofs, they collectively decide whether to discard the prover by raising complaints or recover the missing shares. If t or more verifiers raise complaints, the prover is discarded; otherwise, if fewer than t verifiers complain, the shares are retained, and verifiers with malformed shares or proofs seek to recover missing shares.

The share recovery process is triggered when there are fewer than t complaints against the prover. In this situation, the verifiers holding valid input shares mask their shares with a (VSS) sharing of a random (unknown) value. Next, the verifiers broadcast these masked shares. This allows any verifier with missing shares to recover its share from the masked shares from all the other verifiers and its share of the random (unknown) value. Lastly, we note that the sharing of the random values, utilized as masks in the share recovery process, are generated by the verifiers using any standard Verifiable Secret Sharing (VSS) protocol and can be precomputed in an offline phase. By introducing this share recovery process, the participation of the client is limited to just the first round (to send the share and proof to each of the servers) and not the recovery as is typical in known VSS schemes.

Comparison with Prio [26]. In Prio, one dishonest server can break robustness (although privacy holds against malicious corruption of all but one). Our approach can be extended to the dishonest majority scenario by using an IPS-style MPC-in-the-head approach where the client communication to each server is proportional to size of the circuit, just as in Prio. However, our current work prioritizes guaranteed output delivery, which requires an honest majority. We guarantee cDP, robustness and output delivery against malicious adversaries. Our construction improves upon Prio by reducing the size of proofs sent from the

[8] Prover's time is proportional to generating $O(1)$ proof rather than n proofs.

[9] This generalized polynomial commitment scheme is referred to as a one-to-many zero-knowledge proof in [68]

[10] This is incomparable to the standard one-to-many commit and prove [] where the same relation is used for all the verifiers and all the verifiers have the same output.

client to the server to be sublinear (polylogarithmic) in the circuit size, compared to Prio's linear size. By using only symmetric key-based primitives, our computational costs remain comparable to Prio's.

1.2.2 Differential Privacy.

As with secure aggregation, existing approaches to DP DME face challenges. One approach can be found in the early work by Dwork, Kenthapadi, McSherry, Mironov, and Naor: identify a centrally private solution like the Gaussian mechanism and develop an MPC protocol to sample the required noise in shares [29][11].

Alternatively, we could resort to DP protocols in the local model. In that setting, clients noise their own data before sending them to the output party. The adversary here is able to corrupt all parties except a target client. Local DP protocols avoid the overhead of collective noise generation from the previous approach but are inherently less accurate than centrally private algorithms [9, 18, 43]. Moreover, malicious clients have disproportionate sway on the computation. For example, consider the randomized response protocol developed by Warner for counting [65]. A baseline attack is to perform the local randomization on an adversarial value, but the literature has shown a malicious client can generate a message that introduces greater bias [3, 51, 23]. Attacks against other local DP protocols have also been theoretically and empirically studied [17, 67]. While those results concern specific protocols, Cheu, Smith, and Ullman show how to tailor an attack for *any given* local DP protocol that performs a variant of DME [23].

The approach of wrapping DP around secure aggregation is established in the literature, e.g. work by Kairouz, Liu, & Steinke and by Agarwal, Kairouz, & Liu [40, 2]. We review the approach for completeness. Each client pre-processes their input by introducing a small amount of noise to their data. The noise is discrete Gaussian for Kairouz et al. and Skellam for Agarwal et al. The output party only observes the sum of the noised values. Assuming a bound on the number of malicious clients, the aggregation will have sufficient noise for DP.[12] Post-processing is done to obtain an estimate of the mean with the proper scaling.

Our construction has two significant differences with prior work. First, we utilize input-certified aggregators Π_{Agg} to bound the influence of malicious clients: only values that pass P will be added together. Second, we utilize binomial noise for differential privacy instead of discrete Gaussian or Skellam noise. We quantify the privacy-accuracy tradeoff by way of a novel technical lemma that relates binomial noise and Gaussian noise.

We build intuition here. Assume for now that Π_{Agg} could operate on real-valued numbers. Clients could pre-process their data vectors by adding independent Gaussian noise $\mathbf{N}(0, (\sigma^2/n)I)$, where I is the identity matrix and n is the total number of clients. If all n clients are honest, the aggregated vector observed by the output party has noise distributed as $\mathbf{N}(0, \sigma^2 I)$. It is well-established that this ensures a target parameter ε of differential privacy when $\sigma^2 \approx 1/\varepsilon^2$. If $t > 0$ clients are malicious, the aggregated vector has less variance than targeted. As a result, the effective privacy parameter will not be the same as the target parameter ε but it will follow a smooth function of t, roughly $\varepsilon \cdot \sqrt{n/(n-t)}$.[13]

In reality, we will only be able to operate with finite-precision values. So instead of Gaussian noise with parameter σ^2/n for each coordinate, each client samples binomial noise with parameter h/n which means the aggregate noise in any coordinate is $\mathbf{Bin}(h, 1/2)$. Clients simply need to discretize their data in order to match the discrete nature of binomial noise. We describe this pre-processing in more detail in Subsection 3.2.

We prove that the protocol is private by quantifying how well the binomial approximates the distribution formed by rounding a Gaussian to the nearest integer. The De Moivre-Laplace theorem asserts that, as $h \to \infty$, $\mathbf{Bin}(h, 1/2) - h/2$ approaches $\mathsf{round}(\mathbf{N}(0, \sigma^2 = h/4))$. But it does not tell us how to choose h for a target level of approximation. So we derive a version of De Moivre-Laplace that does: to ensure the binomial is (ε, δ)-close to the rounded Gaussian—where "(ε, δ)-close" is the condition that appears in approximate differential privacy—we require $h \approx \frac{1}{\varepsilon^2} \log \frac{1}{\delta}$ (Theorem 3.4). The upshot is that greater fidelity of approximation requires

[11] A comparison of our work with [29] is discussed in Remark 1.2.2.

[12] For intuition, the reader can assume an honest majority of clients.

[13] Clients could instead sample from $\mathbf{N}(0, (\sigma^2/(n-t_0))I)$ and thereby ensure a target level of privacy for all $t \le t_0$. But the price of such pessimism is larger variance than needed.

a greater scale of noise.

To prevent their signal from being drowned out, clients re-scale their data to match the scale of noise. When there are no malicious clients, the output party obtains an unbiased estimate of the mean by undoing the scaling that the clients performed. The presence of t malicious clients naturally introduces bias but this is bounded by $\tilde{O}(\frac{t}{n} \cdot \frac{\sqrt{d}}{\varepsilon\sqrt{n}})$. This guarantee is made possible by input certification: we only add vectors whose Euclidean norms are below a threshold. The threshold is obtained by bounding the tails of the pre-processing algorithm's distribution.

Section 5 contains a proof that the term $\frac{t}{n} \cdot \frac{\sqrt{d}}{\varepsilon\sqrt{n}}$ in our bias bound is unavoidable for a large class of wrappings around input-certified secure aggregation. The intuition is as follows. Any DP solution for mean estimation (including distributed ones) must have expected squared error $\approx d/\varepsilon^2 n$. If the estimate is unbiased, this is a lower bound on the variance. Now, for a wrapped protocol $\Pi = (\mathsf{PRE}, P, \mathsf{POST})$ where POST is affine, this implies a lower bound on the variance of PRE. The upshot is that malicious clients can contribute an extreme value inside the support of PRE.

Statistical vs. Computational DP. Like prior wrappings around secure aggregation, our proofs of differential privacy operate under the assumption of an ideal functionality that performs secure aggregation. Because the adversary in this setting only views the noisy output of secure aggregation, the wrapping offers protection against even a computationally unbounded adversary. But we will ultimately use a real-world protocol for aggregation whose security guarantees are proved to hold against a bounded adversary. Specifically, we leverage zkSNARKs to efficiently achieve robustness guarantees within our aggregation protocol. While it is indeed possible to construct an information-theoretic protocol with guaranteed output delivery using generic MPC protocols for honest majority settings, such protocols often come with trade-offs. They may require clients to engage in multiple rounds, be communication-intensive, or necessitate a trusted setup. Given these considerations, our protocol opts for efficiency by relying on symmetric-key primitives. Hence, in that case, we are left with the weaker guarantee of computational differential privacy [50]. Refer to Definitions A.1 and A.2 for more detail.

Comparison with "Our Data, Ourselves" (ODO) [29]. This work by Dwork, Kenthapadi, McSherry, Mironov, and Naor [29] identifies a private solution in the central DP model, such as the Gaussian mechanism, and proposes an MPC protocol to sample the required noise in shares. A primary challenge of this generic approach is enabling servers to sample noise in a distributed manner while ensuring simulation-based security. Although ODO offers a method for generating distributed noise, it does not establish the necessary simulation-based security.

To achieve robustness, ODO employs interactive MPC, whereas our approach utilizes more efficient zkSNARKs. Additionally, ODO requires each server to perform VSS on three values and conduct multiple interactive checks to generate noise shares. In contrast, our clients add noise locally, which is verified through zkSNARKs, leading to a significant reduction in communication overhead and the number of rounds required. Furthermore, ODO treats all parties equally, involving both clients and servers in the computation and not minimizing client participation. In our protocol, however, we require only a single round of client involvement. Overall, our approach reduces the number of VSS sharings, improves the efficiency of robustness checks, and supports more generic robustness checks.

2 Security Defintion

We introduce some notation to describe our multiparty computation protocols. In this work, we only consider static corruptions, i.e. the adversary needs to decide which party it corrupts before the execution

begins. We use the following security parameters in our definition. We denote by κ a computational security parameter and by ϵ, δ statistical security parameters that captures statistical errors. Let $\Pi = \langle P_1, P_2, \ldots, P_n \rangle$ denote a n-party protocol, where each party is given an input (x_i for P_i) and security parameters 1^s and 1^κ. We allow honest parties to be PPT in the entire input length (this is needed to ensure correctness when no party is corrupted) but bound adversaries to time $\mathsf{poly}(\kappa)$ (this effectively means that we only require security when the input length is bounded by *some* polynomial in κ). We denote by $\mathbf{EXEC}_{\Pi,\mathcal{I},\mathcal{A}(z)}(x_1, \ldots, x_n, \kappa, \epsilon, \delta)$ the random variable representing an execution of Π with adversary $\mathcal{A}$ controlling parties in $\mathcal{I}$, where z is the auxiliary input, x_i is P_i's initial input, κ is the computational security parameter and s is the statistical security parameter. We denote by $\mathsf{out}_{P_i}(e)$ and $\mathsf{VIEW}_{\mathcal{A}}(e)$ as functions that take an input an instance of the execution e and outputs the output of the honest party P_i and view of the adversary $\mathcal{A}$ respectively in the execution e. In some of our protocols the parties have access to ideal model implementation of certain cryptographic primitives $\mathcal{G}$ which we refer as a protocol in the $\mathcal{G}$-hybrid. The definitions below extend to protocols in $\mathcal{G}$-hybrid. The *canonical protocol* in the $\mathcal{G}$ hybrid described the protocol in which all honest parties take their inputs and send them to $\mathcal{G}$ and finally output whatever $\mathcal{G}$ returns.

Definition 2.1. *(DME-Usefulness).* Given $X = (x_1, \ldots, x_n)$, let μ_X be the sample mean of a distribution $\mathcal{D}$ i.e., $\mu_X \leftarrow \frac{1}{n}\sum_{i\in[n]} x_i$ where x_i are sampled from $\mathcal{D}$. We say that the protocol Π satisfies $\nu(t)$-DME-usefulness w.r.t $\mathcal{D}$ if for any adversary $\mathcal{A}$ controlling up to t clients and P_n receiving the output,

$$\mathbb{E}\left[\left\{x_i \leftarrow \mathcal{D}; e \leftarrow \mathbf{EXEC}_{\Pi,\mathcal{I},\mathcal{A}(z)}(x_1, \ldots, x_n, \kappa, \epsilon, \delta) : \left\|\mathsf{out}_{P_n}(e) - \mu_X\right\|_2^2\right\}\right] \leq \nu(t)$$

Definition 2.2. A protocol Π is said to satisfy (ϵ, δ)-IND-CDP if it holds that the randomized algorithm, on input vector $(x_1, \ldots, x_n)$, samples an execution e of Π with adversary $\mathcal{A}$ where P_i's input is set to x_i and outputs $\mathsf{VIEW}_{\mathcal{A}}(e)$ offers (ϵ, δ)-approximate computational differential privacy (see Definition A.2).

Definition 2.3. A protocol Π is said to be (ϵ, δ, ν)-Differentially-Private Distributed Mean Estimation protocol (DP-DME) for distribution $\mathcal{D}$ if it satisfies the following two properties: (1) (ϵ, δ)-IND-CDP and (2) ν-*DME-Usefulness* w.r.t $\mathcal{D}$.

Remark 2.4 (Comparison with simulation-based security)**.** *Our security definition captures correctness and privacy through game-based definitions of usefulness and differential privacy. While we gain efficiency, this is strictly weaker from a privacy standpoint than an MPC protocol that realizes, via the standard simulation-based security notion, an ideal functionality that implements (the same) differentially private mechanism with robustness (i.e., input certification) to compute the mean. This is because the latter, in addition to implying (ϵ, δ)-IND-CDP, would also guarantee that the view is simulatable from the output and inputs of the corrupted parties. In fact, we demonstrate that our protocol satisfies these game-based properties by considering a slightly weaker ideal functionality (which is sufficient for the guarantees) and securely realizing it via an MPC protocol in the standard simulation-based paradigm.*

3 Differentially-Private Distributed Mean Estimation

Basic notations. We denote the security parameter by λ. We say that a function $\mu : \mathbb{N} \to \mathbb{N}$ is *negligible* if for every positive polynomial $p(\cdot)$ and all sufficiently large λ's it holds that $\mu(\lambda) < \frac{1}{p(\lambda)}$. We use the abbreviation PPT to denote probabilistic polynomial-time and denote by $[n]$ the set of elements $\{1, \ldots, n\}$ for some $n \in \mathbb{N}$. $\mathcal{BC}(\cdot)$ denotes the communication over the broadcast channel.

We will use $\mathsf{n_c}, \mathsf{t_c}$ to denote the total number of clients and the number of malicious clients, respectively, but will drop the subscript when we describe how we enforce DP in Sections 3 and 5 (but keep the subscripts' in Section 4). This is because we focus solely on client behavior in those sections, so there is no need to

disambiguate clients from servers. Tables 2 and 3 list the variables used in this paper along with their definitions.

Distributed Mean Estimation via Wrapped Protocols. In this section, we will describe our differentially-private mechanism modularly by assuming that the parties i.e., clients $\mathcal{U} = \{\mathcal{U}_1, \ldots, \mathcal{U}_{\mathsf{n_c}}\}$, have access to a input-certified secure aggregation (in short, certified aggregation) ideal functionality $\mathcal{F}_{\mathsf{Agg}}$, described in Figure 4. Specifically, we construct a "wrapping" around the certified aggregation functionality, which consists of $(\mathsf{PRE}, P, \mathsf{POST})$, where PRE represents a randomized pre-processing algorithm, P is the input-certification predicate, and POST is the post-processing algorithm. The wrapped protocol operates as follows.

1. Each client $\mathcal{U}_i$ with input X_i pre-processes its inputs as $Y_i = \mathsf{PRE}(X_i)$.
2. The certified aggregation functionality $\mathcal{F}_{\mathsf{Agg}}$ is invoked on the pre-processed inputs $\{Y_i\}_{\mathcal{U}_i \in \mathcal{U}}$ using P as the certification predicate.
3. The output party receives the aggregated result Y_{agg} from the $\mathcal{F}_{\mathsf{Agg}}$ functionality, which is subsequently post-processed using POST and designated as the final output.

Differential Privacy for Wrappings. The classic notion of differential privacy makes the implicit assumption that the privacy adversary only has access to the output of an algorithm. In our setting, the adversary has a view that potentially contains more information than the outcome of input-certified aggregation.

Definition 3.1. A wrapping Π around certified secure aggregation functionality satisfies (ε, δ)-computational differential privacy if the algorithm that generates the adversary's view jointly with the protocol's output satisfies (ε, δ)-computational differential privacy.

We now focus on instantiating the wrapping $\Pi = (\mathsf{PRE}, P, \mathsf{POST})$. Depending on how we instantiate Π, we obtain different variants of our protocol. In this regard, we propose a specific variant of our protocol, referred to as Distributed Binomial Mechanism (DBM). As the name suggests, each client in the DBM adds binomial noise to a discretized version of their input (i.e. PRE). Subsequently, the clients invoke an instance of certified aggregation on the pre-processed inputs. This calls for transforming the input-certification predicate over the actual inputs to a new predicate over the pre-processed inputs. This new predicate, say P, should not exclude any honest clients who honestly generate and pre-process their inputs. The output party's view is the outcome of a DP algorithm because the aggregate vector has noise sourced from all honest clients. The view is usable as a mean estimate, after a post-processing step (i.e., POST).

We compare our DBM with other instantiation of wrappings in Table 1. These include analysis of a different wrapping around Π_{Agg} by Chen, Özgür, and Kairouz [19]. Unlike the DBM, their Poisson-Binomial mechanism (PBM) requires shared randomness. More significantly, small constants are challenging to obtain in the version that Chen et al. explicitly give ("Kashin-PBM" in Table 1). We are able to obtain a better leading constant for a variant but cannot avoid an extra logarithmic term ("Rotation-PBM"). Refer to Appendix G for more detail.

Observe that all of our upper bounds on mean-squared error have a $\frac{t^2}{n^2} \cdot \frac{d}{n}$ term. In Section 5, we prove that this term is unavoidable for a natural class of protocols that wrap around certified aggregation.

For simplicity, we use n to denote the total number of clients and t to denote the number of corrupt clients (rather than $\mathsf{n_c}$ and $\mathsf{t_c}$ respectively).

3.1 Building Blocks: Binomial and Gaussian Mechanisms

Let f be any Δ-sensitive function over the integers $\mathbb{Z}^d$, meaning that $\|f(W) - f(W')\|_2 \leq \Delta$ for matrices W, W' differing on one row. Let $\mathsf{BM}_{f,m}$ be the algorithm that takes some W as input and reports $f(W) + \eta$

	D.P. Type	Field Size	Shared Randomness	Bound on Expected ℓ_2^2 error with t corruptions
DBM **(our work)**	(ε, δ) Approx.	$\tilde{O}\Big(nd \log \frac{d}{\delta} + \frac{d^2}{\varepsilon^2} \log^2 \frac{d}{\delta}\Big)$	None	$O\Big(\frac{t^2}{n^2} \cdot \frac{d}{\varepsilon^2 n} \log^2 \frac{e^\varepsilon nd}{\delta} + \frac{n-t}{n} \cdot \frac{d}{\varepsilon^2 n^2} \log \frac{e^\varepsilon}{\delta}\Big)$ Theorem 3.6
Kashin-PBM by Chen et al. [19]	(α, ε) Rényi	$n \cdot \ell$	$d \times d$ matrix	$O\Big(\frac{t^2}{n^2} \cdot \frac{\alpha \ell d}{\varepsilon n} + \frac{n-t}{n} \cdot \frac{\alpha d}{\varepsilon n^2}\Big)$ Theorem G.9
Rotation-PBM implicit in [19]			$O(d)$ angles	$O\Big(\log(nd) \cdot \Big(\frac{t^2}{n^2} \cdot \frac{\alpha \ell d}{\varepsilon n} + \frac{n-t}{n} \cdot \frac{\alpha d}{\varepsilon n^2}\Big)\Big)$ Theorem G.14
Gaussian Mech. (centralized DP [30])	Approx.	N/A	N/A	$O\Big(\frac{t^2}{n^2} + \frac{d}{\varepsilon^2 n^2} \log \frac{1}{\delta}\Big)$
	Rényi	N/A	N/A	$O\Big(\frac{t^2}{n^2} + \frac{d\alpha}{\varepsilon n^2}\Big)$

Table 1: Comparison of solutions for mean estimation of data in $\mathcal{B}^d$. "Centralized DP" refers to the setting where a single party receives all client data in the clear. We assume $d \geq n$ to simplify some bounds. The value $\ell \in \mathbb{N}$ is a positive integer parameter of the Poisson-Binomial Mechanism (PBM). Refer to Appendix G for more information on the PBM and Rényi DP.

where $\eta[j] \sim_{\text{i.i.d.}} \mathbf{Bin}(m, 1/2) - h/2$ for all $j \in [d]$. Let the Gaussian mechanism GM_{f,σ^2} be the algorithm that reports $f(W) + \eta$ where $\eta[j] \sim_{\text{i.i.d.}} \mathbf{N}(0, \sigma^2)$ for all $j \in [d]$.

The Gaussian mechanism is a classic building block for differentially private algorithms. We reproduce one statement of its privacy guarantees below.

Theorem 3.2 (From Dwork & Roth [31])**.** *Fix $\varepsilon, \delta \in (0,1)$. If $\sigma^2 \geq \frac{2\Delta^2}{\varepsilon^2} \ln \frac{5}{4\delta}$, then GM_{f,σ^2} is (ε, δ)-differentially private.*

Due to the discrete nature of computation and communication, we will deal with a discrete version of the Gaussian mechanism. Let $\widetilde{\mathsf{GM}}_{f,\sigma^2}$ be the algorithm that reports $f(W) + \mathsf{round}(\eta)$, where round is the function that rounds a real number to the nearest integer and $\eta[j] \sim_{\text{i.i.d.}} \mathbf{N}(0, \sigma^2)$ for all $j \in [d]$.

Rounding a real-valued η before adding it to an integer is interchangable with rounding the sum of η and that same integer. Hence,

Lemma 3.3. *On any input W, $\widetilde{\mathsf{GM}}_{f,\sigma^2}$ is identically distributed with $\mathsf{round} \circ \mathsf{GM}_{f,\sigma^2}$.*

Now, we show that the noise in $\widetilde{\mathsf{GM}}_{f,\sigma^2}$ is well-approximated by binomial noise with the same variance ($\sigma^2 = h/4$). The textbook De Moivre-Laplace theorem asserts this is the case as the variance approaches infinity, but we require a way to quantify the distance between the distributions for a finite value of h. This is done in the following theorem.

Theorem 3.4 (Finite-value De Moivre-Laplace)**.** *Fix any $\varepsilon \in (0,1)$, $\delta \in (0, 2e^{-6})$ and any even $m > \frac{12}{\varepsilon^2} \ln^2 \frac{2}{\delta}$.*

$$\mathsf{round}(\mathbf{N}(0, \sigma^2 = m/4)) \approx_{\varepsilon,\delta} \mathbf{Bin}(m, 1/2) - m/2$$

A proof can be found in Appendix C. Now we can transfer the privacy analysis of the Gaussian mechanism to the binomial mechanism: $\widetilde{\mathsf{GM}}_{f,\sigma^2}$ inherits the privacy of GM_{f,σ^2} by closure under post-processing and $\mathsf{BM}_{f,m}$ inherits the privacy of $\widetilde{\mathsf{GM}}_{f,\sigma^2}$ via Theorem 3.4.

Theorem 3.5 (Privacy of BM)**.** *Let f be any Δ-sensitive function over the integers $\mathbb{Z}^d$, let W, W' be matrices differing on one row. Fix any ε_0, ε_1, δ_0, and δ_1 in the interval $(0,1)$. If m is an even integer such that $m > \max\Big(\frac{12}{\varepsilon_0^2} \ln^2 \frac{2}{\delta_0}, \frac{8\Delta^2}{\varepsilon_1^2} \ln \frac{5}{4\delta_1}\Big)$, then for any event E,*

$$\begin{aligned}\mathbb{P}\Big[\mathsf{BM}_{f,m}(W) \in E\Big] \leq \exp(2d\varepsilon_0 + \varepsilon_1) \cdot \mathbb{P}\Big[\mathsf{BM}_{f,m}(W') \in E\Big] \\ + 2\exp(d\varepsilon_0 + \varepsilon_1) \cdot d\delta_0 + \exp(d\varepsilon_0) \cdot \delta_1\end{aligned}$$

A proof can be found in Appendix D. For any fixed $\varepsilon, \delta \in (0,1)$, it is of course possible to choose the parameters in such a way that BM satisfies (ε, δ)-differential privacy. But we leave the theorem in its four-parameter state to ease analysis further down the line.

3.2 Distributed Binomial Mechanism (DBM) for DP-DME

We are now ready to present the distributed binomial mechanism (DBM) for differentially private mean estimation. A "wrapping" around certified aggregation, $\Pi_{g,b,\tau,r}$ is specified by a randomized pre-processing algorithm $\mathsf{PRE}_{g,b,\tau}$, a predicate inBall_r, and a post-processing function POST_g.

1. Honest clients run $\mathsf{PRE}_{g,b,\tau}$ locally to generate inputs to the certified aggregator. It performs two main operations. First, it maps the client's data—a vector of continuous values—to a vector of discrete values. g is a parameter that determines granularity. Second, it generates a vector of independent binomial noise: each entry is the number of heads after b tosses of a fair coin. The client's input for Σ is the sum of the two vectors. If a party added all n of these sums, the output would be identical to an execution of the binomial mechanism, which we showed is differentially private for the right parameters.

2. inBall_r is the predicate that takes any client vector $Y_i \in \mathbb{F}_q^d$ as input and outputs 1 if and only if it is inside the ball of radius r ($\|Y_i\|_2 \leq r$).

3. The output party's post-processing function POST_g produces an estimate of the mean by compensating for discretization: $\mathsf{POST}_g(y_{\text{agg}}) = \frac{2}{ng} \cdot y_{\text{agg}}$

Algorithm 1: Local pre-processor $\mathsf{PRE}_{g,b,\tau}$ for DBM

Input: $X_i \in \mathcal{B}^d$
Output: $Y_i \in \mathbb{F}^d$
`/* Step 1: map each coordinate to integer` $\in [-g/2, +g/2]$`... */`
For $j \in [d]$
  $\hat{X}_i[j] \leftarrow g \cdot \frac{X_i[j]}{2}$
  $W_i[j] \leftarrow \lfloor \hat{X}_i \rfloor + \mathbf{Ber}(\hat{X}_i - \lfloor \hat{X}_i \rfloor)$
`/* Step 2: sample binomial noise */`
For $j \in [d]$
  $\eta_i[j] \sim \mathbf{Bin}(b, 1/2) - b/2$
`/* Step 3: Truncate noise vector if overly large */`
If $\left\|\eta_i\right\|_2 > \tau$ **:**
  $\eta_i \leftarrow \vec{0}$
Return $Y_i \leftarrow W_i + \eta_i$

We give pseudocode for $\mathsf{PRE}_{g,b,\tau}$ in Algorithm 1. Note that it has a truncation step: if the noise vector exceeds a norm bound τ, the randomizer resets it to zero. While this slightly complicates our privacy analysis, it allows the aggregator to filter out overly large vectors sent by malicious clients.

This filtering defends against manipulation attacks (see Figure 5). Each attack K is specified by a set of corrupt clients $\mathcal{C} \subset [n]$ (with $|\mathcal{C}| = t$) and a joint distribution $\mathbf{T}$ over $\{y \in \mathbb{F}_q^d \mid \left\|y\right\|_2 \leq r\}^t$. The malicious clients send messages $\{y_i^K\}_{i \in \mathcal{C}}$ sampled from $\mathbf{T}$; due to the certified aggregation, the malicious clients are only able to contribute vectors in a restricted set. While they may deviate from $\mathsf{PRE}_{g,b,\tau}$, we prove that the error they introduce will be bounded.

Theorem 3.6. *For $\varepsilon \in (0, 9/10)$ and $\delta \in (0, 2e^{-6})$, there are choices of parameters g, b, τ, r such that $\Pi_{g,b,\tau,r} = (\mathsf{PRE}_{g,b,\tau}, \mathsf{inBall}_r, \mathsf{POST}_g)$ ensures (ε, δ)-DP and solves DME with variance $O(\frac{d}{\varepsilon^2 n^2} \log \frac{e^\varepsilon}{\delta})$ when there are no malicious clients. When under any attack by $t \le n/6$ clients, it ensures $\left(\varepsilon \cdot \sqrt{\frac{n}{n-t}},\ \delta \cdot \exp\left(\varepsilon \cdot \sqrt{\frac{n}{n-t}} - \varepsilon\right)\right)$-differential privacy and solves DME with mean-squared error*

$$\frac{t^2}{n^2} \cdot O\left(1 + \frac{\sqrt{d}}{\varepsilon\sqrt{n}} \log \frac{e^\varepsilon n d}{\delta}\right)^2 + \frac{n-t}{n} \cdot O\left(\frac{d}{\varepsilon^2 n^2} \log \frac{e^\varepsilon}{\delta}\right)$$

Bias is only present due to attacks. It suffices for certified aggregation to operate on numbers in a field of size

$$n \cdot (g + b) = \tilde{O}\left(nd \log \frac{d}{\delta} + \frac{d^2}{\varepsilon^2} \log^2 \frac{d}{\delta}\right)$$

We note that the $t \le n/6$ condition is arbitrary: any other constant fraction bound can be substituted, at the cost of adjusting the range of ε. Appendix D contains a proof of the theorem. The proof will depend on sub-claims that the rest of this section will elaborate. Proofs of these sub-claims are also provided in the appendix. Also, we visualize the error bound in Figure 6 of Appendix D, in a regime where $d < n$ and a regime where $d > n$.

Remark 3.7 (Servers Adding Noise)**.** *An alternative to clients adding noise to their inputs, we could consider the servers adding the noise by simply using our PRE algorithm with an all zero input. However, this requires both servers and clients to submit proofs as we still need robustness, whereas our protocol only requires client proofs. The upside when the servers add noise is that we can obtain slight improvements in the mean-squared error since we assume strong honest-majority among servers (i.e., $n \ge 3t + 1$). The downside is that such an approach is mainly applicable when PRE adds "additive" noise to clients' input, as in our work for DPDME. For more generic functions applied by PRE, as in the work of Chen et al. [1], incorporating server-added noise is non-trivial (need distributed noise generation mechanism), but our approach remains applicable. However, we leave it as future work to further analyze such an approach.*

3.2.1 Properties of Pre-Processing Algorithm

In order to analyze the protocol's privacy and accuracy guarantees, we first establish key properties of the pre-processing algorithm $\mathsf{PRE}_{g,b,\tau}$. First, its output is a re-scaled but unbiased representation of the input vector:

Lemma 3.8. *For any $X_i \in \mathcal{B}^d$, if $Y_i \leftarrow \mathsf{PRE}_{g,b,\tau}(X_i)$, then $\mathbb{E}[Y_i[j]] = \frac{g}{2} \cdot X_i[j]$*

Second, the output vector lies in a ball of predetermined size:

Lemma 3.9. *For any $X_i \in \mathcal{B}^d$, if $Y_i \leftarrow \mathsf{PRE}_{g,b,\tau}(X_i)$, then $\|Y_i\|_2 \le g/2 + \sqrt{d} + \tau$ with probability 1.*

An immediate consequence of this result is that we can assign $r \leftarrow g/2 + \sqrt{d} + \tau$ and thereby bound the influence of malicious clients. Similar algebra also implies a bound on the ℓ_2 sensitivity of W_i, W_i', the discrete representation of X_i, X_i':

Lemma 3.10. *For any $X_i, X_i' \in \mathcal{B}^d$, if we construct $W_i[j], W_i'[j]$ according to $\mathsf{PRE}_{g,b,\tau}$ for every $j \in [d]$, then $\left\|W_i - W_i'\right\|_2 \le g + 2\sqrt{d}$ with probability 1.*

Finally, we can bound the maximum magnitude of any entry in the output vector. This will allow us to determine the field size in an instantiation of Π_{Agg}.

Lemma 3.11. *For any $X_i \in \mathcal{B}^d$, if $Y_i \leftarrow \mathsf{PRE}_{g,b,\tau}(X_i)$, then $\|Y_i\|_\infty \le g/2 + b/2$ with probability 1.*

3.2.2 Guarantees of Protocol

We will first bound the error of the DBM's mean estimate in terms of the parameters g, b, τ:

Lemma 3.12 (Mean Squared-Error of DBM). *Fix any attack K for t malicious clients. If $\mu^K \leftarrow \Pi^K_{g,b,\tau,r}(X)$, then the mean squared-error of μ^K with respect to $\mu \leftarrow \frac{1}{n}\sum_{i\in[n]} X_i$ is*

$$\mathbb{E}\left[\left\|\mu^K - \mu\right\|_2^2\right] \leq \frac{4t^2}{n^2} \cdot (1 + \sqrt{d}/g + \tau/g)^2 + \frac{d}{n^2} \cdot \frac{h(b+1)}{g^2}$$

When all clients are honest ($h = n$, $t = 0$), this simplifies to $\frac{d}{4ng^2} \cdot (b+1)$

It remains to identify parameter choices that guarantee differential privacy. Specifically, we will choose g, b, τ such that the process that generates the output party's view—which we denote as $\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}^K_{g,b,\tau}(X)$—satisfies DP. The privacy parameters will depend on the number of malicious clients but in the ideal case where there is no attack—which we denote as $\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau}(X)$—the parameters will equal a target (ε, δ) pair.

As mentioned previously, we will argue that $\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}^K_{g,b,\tau}(X)$ simulates the binomial mechanism $\mathsf{BM}(X)$ and then invoke the privacy analysis of $\mathsf{BM}(X)$ (Theorem 3.5). One small hurdle in our analysis is that our pre-processing algorithm $\mathsf{PRE}_{g,b,\tau}$ truncates noise that exceeds norm τ. This means even an execution without attack will not be identical to the output of the binomial mechanism $\mathsf{BM}(X)$. Hence, we cannot immediately invoke Theorem 3.5. But we can bound the statistical distance between $\mathsf{PRE}_{g,b\tau}$ and $\mathsf{PRE}_{g,b,\infty}$, which never resets the noise it samples to $\vec{0}$.

Lemma 3.13. *If $\tau \geq \sqrt{\frac{db}{2}\ln\frac{2nd}{\delta}}$, then for any input X, $\mathsf{PRE}_{g,b,\infty}(X)$ is within statistical distance δ of $\mathsf{PRE}_{g,b,\tau}(X)$*

Proof. We use Hoeffding's inequality to bound the probability that any noise sample $\eta_i[j]^2$ exceeds $\frac{b}{2}\ln\frac{2nd}{\delta}$ by δ/nd. A union bound over all clients and dimensions completes the proof. □

We can factor this slack into the analysis from the preceding section and then choose g, b, τ such that $\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau}(X)$ satisfies (ε, δ)-DP.

Lemma 3.14 (Parameters for DBM's Privacy). *Fix $t \leq n/6$, $\varepsilon \in (0, 9/10)$, and $\delta \in (0, 2e^{-6})$. Let $\varepsilon_{\text{priv}} \leftarrow 99\varepsilon/100$, $\varepsilon_{\text{sim}} \leftarrow \varepsilon/200d$, $\delta_{\text{priv}} \leftarrow \delta/5e^{\varepsilon}$, and $\delta_{\text{sim}} \leftarrow \delta/5de^{\varepsilon}$. When $b \geq \frac{12}{n\varepsilon_{\text{sim}}^2}\ln^2\frac{2}{\delta_{\text{sim}}}$, $g \leq \varepsilon_{\text{priv}}\sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}} - 2\sqrt{d}$, and $\tau \geq \sqrt{\frac{db}{2}\ln(2nd/\delta_{\text{priv}})}$, $\Pi_{g,b,\tau,r}$ is $\left(\varepsilon \cdot \sqrt{\frac{n}{h}},\ \delta \cdot \exp\left(\varepsilon \cdot \sqrt{\frac{n}{h}} - \varepsilon\right)\right)$-differentially private. When all clients are honest ($h = n$), this simplifies to (ε, δ)-DP.*

Appendix D contains a proof that lemmas 3.12 & 3.14 collectively imply Theorem 3.6. We provide a privacy accounting analysis when the protocol is repeated R times in Appendix E.

3.3 Protocol and Main Theorem

Consider a protocol (i.e., wrapping) $\Pi' = (\mathsf{PRE}, P, \mathsf{POST})$ in the $\mathcal{F}_{\mathsf{Agg}}$-hybrid: the clients pre-process their inputs as per PRE and send it to $\mathcal{F}_{\mathsf{Agg}}$. Upon receiving the output from $\mathcal{F}_{\mathsf{Agg}}$, send this output to the output party, who then post-processes it as per POST and sets it as the output. Let Π_{DBM} be the protocol obtained by taking Π' and replacing all calls to the $\mathcal{F}_{\mathsf{Agg}}$-functionality with our input-certified secure aggregation protocol Π_{Agg}. For completeness, our main theorem includes the costs of our secure aggregation protocol, with details in Section 4. We defer the proof of the main theorem to Appendix M.

Theorem 3.15 (Main Theorem). *Let $\mathsf{n}_s, \mathsf{t}_s \in \mathbb{N}$ such that $\mathsf{t}_s < \mathsf{n}_s/3$. Given a distribution $\mathcal{D}$ with support contained in a Euclidean unit ball, Protocol Π_{DBM} is a (ϵ, δ, ν)-Differentially-Private Distributed Mean Estimation (*DP-DME*)*

protocol for $\mathcal{D}$ among $\mathsf{n_c}$ clients each holding input vectors of length d with elements in some finite field $\mathbb{F}$, n_s servers, and an output party $\mathcal{O}$, which is secure against a static, malicious rushing adversary that can maliciously corrupt an arbitrary number of clients, up to $\mathsf{t_s}$ servers and the output party and ensures guaranteed output delivery. This protocol achieves (ε, δ)-DP and solves DME with variance $O(\frac{d}{\varepsilon^2 n^2} \log \frac{e^\varepsilon}{\delta})$ when there are no malicious clients. When under any attack by $t \leq n/6$ clients, it ensures $\left(\varepsilon \cdot \sqrt{\frac{n}{n-t}},\ \delta \cdot \exp\left(\varepsilon \cdot \sqrt{\frac{n}{n-t}} - \varepsilon\right)\right)$-differential privacy and solves DME with mean-squared error

$$\frac{t^2}{n^2} \cdot O\left(1 + \frac{\sqrt{d}}{\varepsilon\sqrt{n}} \log \frac{e^\varepsilon n d}{\delta}\right)^2 + \frac{n-t}{n} \cdot O\left(\frac{d}{\varepsilon^2 n^2} \log \frac{e^\varepsilon}{\delta}\right)$$

Bias is only present due to attacks. It suffices for certified aggregation to operate on numbers in a field of size

$$n \cdot (g + b) = \tilde{O}\left(nd \log \frac{d}{\delta} + \frac{d^2}{\varepsilon^2} \log^2 \frac{d}{\delta}\right)$$

Moreover, the total communication complexity between a client and each server is $O(d + \mathbb{d} \log |P| + \log^2(\mathsf{n}_s \cdot d))$ field elements where $|P|$ and $\mathbb{d}$ are the size and depth (respectively) of the circuit associated with the predicate P. The total amortized communication among the servers is $O(\mathsf{n_c} \cdot d \cdot \mathsf{n}_s^3)$ field elements in the offline phase and $O(\mathsf{n_c} \cdot d \cdot \mathsf{n}_s^2)$ field elements in the online phase for a sufficiently large d. Additionally, a client is required to engage in only a single round of communication.

4 Input-Certified Secure Aggregation

We present our main protocol for Input-Certified Secure Aggregation in this section. We provide a brief overview and then provide our constructions for Distributed Commit-and-Prove (dCP) and Verifiable Relation Sharing (VRS).

At a high level, we identify Verifiable Relation Sharing (VRS) introduced in the work of Applebaum et al. [4] as the key tool for robust secure aggregation. Recall that VRS with respect to some predicate P allows a dealer to share a secret to a set of servers with the guarantee that honest servers receive valid shares of a secret that satisfies the predicate P or rejects the dealer. Given a VRS scheme an input-certified secure aggregation proceeds as follows: Each client acting as the dealer shares its secret using the VRS to the servers. The honest servers at the end of the sharing phase aggregate their shares and deliver the output to the output party that reconstructs the final result. In order to achieve guaranteed output delivery, the secret shares need to satisfy an error-recovery mechanism (obtained using standard error-correcting code based sharing) so that the output party can reconstruct the final result even if the malicious server provide incorrect values to the output party.

A key contribution of this work is to design a VRS scheme that meets our requirement that the client speaks only once and is concretely efficient. Toward this, we first design a VSS scheme inspired by the proactive secret sharing scheme [39, 38] and incorporate input certification relying on an extension of the distributed polynomial commitment scheme of Zhang et al. [68]. We abstract this primitive as distributed commit and prove (dCP). We next describe our dCP and VRS protocols.

4.1 Distributed Commit-and-Prove (dCP)

We present our distributed commit-and-prove (dCP) protocol, which involves a prover and n verifiers. Given n relations $\mathcal{R}_1, \ldots, \mathcal{R}_n$, the prove, with input w, aims to prove to each verifier $\mathcal{V}_i$ that (x_i, w) belongs to $\mathcal{R}_i$ for each $i \in [n]$. Our focus is on designing a dCP protocol for specific types of relations, which we specify via a circuit. First, we define what it means for a relation to be *specified* via a circuit, and then we detail the types of circuits for which we will construct a dCP.

Functionality $\mathcal{F}_{\mathsf{dCP}}$

$\mathcal{F}_{\mathsf{dCP}}$ runs among the Prover $\mathcal{P}$ and n Verifiers $\mathcal{V} = \{\mathcal{V}_1, \ldots, \mathcal{V}_n\}$ and an adversary Sim. It is parameterized by n relations $(\mathcal{R}_1, \ldots, \mathcal{R}_n)$:

- **Commit Phase**: Upon receiving a message $(\mathsf{commit}, \mathsf{sid}, \mathcal{P}, w)$ from the prover $\mathcal{P}$, record the values w and $\mathcal{P}$, and send the message $(\mathsf{receipt}, \mathsf{sid}, \mathcal{P})$ to the verifiers in $\mathcal{V}$ and Sim. (If a commit message has already been received, then ignore any other messages with the same sid.)
- **Prove Phase**: Upon receiving a message $(\text{dCP-prover}, \mathsf{sid}, x_j, \mathcal{V}_j)$ from the prover $\mathcal{P}$ compute $\mathcal{R}_j(x_j, w)$. If $\mathcal{R}_j(x_j, w) = 1$, then send the message $(\text{dCP-proof}, \mathsf{sid}, x_j, \mathsf{accept})$ to the verifier $\mathcal{V}_j$ and $\mathcal{S}$, and $\mathcal{V}_j$ outputs (x_j, accept). Otherwise, send $(\text{dCP-proof}, \mathsf{sid}, \mathsf{reject})$ to the verifier $\mathcal{V}_j$ and $\mathcal{V}_j$ outputs reject.

Figure 1: Ideal Functionality for Distributed Commit-and-Prove

Definition 4.1 (Relation Specification via Circuits)**.** Consider a circuit $\mathsf{C} : \mathbb{F}^{\deg+1} \to \mathbb{F}^n$ that receives an input $\mathbf{in}$ and outputs $(\mathbf{out}_1, \ldots, \mathbf{out}_n)$, where $\mathbf{out}_j = [C(\mathbf{in})]_j$ is the j^{th} output of the circuit given input $\mathbf{in}$. A pair $(\mathbf{out}_j, \mathbf{in})$ *belongs to* $\mathcal{R}_j$ if the evaluation by $\mathsf{C}(\mathbf{in})$ does not output $(\perp, \ldots, \perp)$. Furthermore, we say that the distributed relation $(\mathcal{R}_1, \ldots, \mathcal{R}_n)$ *is specified by* a circuit C if $(\mathbf{out}_j, \mathbf{in})$ *belongs to* $\mathcal{R}_j$ for each $j \in [n]$.

In this work, we consider specific types of circuits that take as input a polynomial and output the evaluation of the polynomials at predetermined points, depending on whether the input to the circuit satisfies certain criteria. Formally, consider a circuit $\mathsf{C} : \mathbb{F}^{\deg+1} \to \mathbb{F}^n$ that receives an input $\mathbf{in}$, which is a polynomial f of degree deg and outputs $(\mathbf{out}_1, \ldots, \mathbf{out}_n)$, where each $\mathbf{out}_j = f(\alpha_j)$ provided that $P(f(0)) = 1$; if not, $\mathbf{out}_j$ is set to $\perp$ for every $j \in [n]$. Here, $P : \mathbb{F} \to \{0, 1\}$ denotes a predicate, and $\alpha_1, \ldots, \alpha_n$ represent predetermined evaluation points. The circuit C comprises two parts: (1) to check if $P(f(0) = 1$ and (2) evaluate the polynomial f at n evaluation points. The latter portion of the circuit can be instantiated via the butterfly circuit for the FFT algorithm, similar to [68].

Our Distributed Commit-and-Prove protocol is based on the transparent polynomial commitment scheme given by Zhang et. al. [68]. . The ideal functionality for Distributed Commit-and-Prove (dCP), represented by $\mathcal{F}_{\mathsf{dCP}}$, is provided in Figure 1. The corresponding protocol Π_{dCP}, which securely implements $\mathcal{F}_{\mathsf{dCP}}$, is detailed in Figure 9 within Appendix J. The formal theorem is presented below, with a sketch of the proof provided in Appendix J.

Theorem 4.2. *Given $(\mathcal{R}_1, \ldots, \mathcal{R}_n)$, which are specified by a circuit $C : \mathbb{F}^{t+1} \to \mathbb{F}^n$ in accordance with Definition 4.1, the Π_{dCP} protocol involving a prover $\mathcal{P}$ and n verifiers $\mathcal{V}_1, \ldots, \mathcal{V}_n$, securely realize the $\mathcal{F}_{\mathsf{dCP}}$ functionality against a static malicious adversary $\mathcal{A}$ who controls at most t verifiers. The complexities are as follows:*

- *The prover's time is $O(|\mathsf{C}| + n \log n)$.*
- *Each verifier's time is $O(\mathbb{d} \log |\mathsf{C}| + \log^2 n)$.*
- *The proof size is $O(\mathbb{d} \log |\mathsf{C}| + \log^2 n)$ field elements for each verifier.*

Here, $\mathbb{d}$ represents the depth of the circuit C.

Efficiency. The above complexities follow from Theorem 3 of [68] where $t = O(n)$. For simplicity, we have considered the case where the prover's input is a single field element, $x \in \mathbb{F}$. To extend this to the case where the prover's input is a vector, say $X \in \mathbb{F}^d$, we consider a larger circuit $C_d : \mathbb{F}^{d \cdot (t+1)} \to \mathbb{F}^{d \cdot n}$ which takes as input d t-degree polynomial and outputs evaluations of each of the d polynomials at n points. The above costs can be adapted for d input vector by considering C_d (instead of circuit C).

4.2 Verifiable Relation Sharing (VRS) from dCP

We present our verifiable relation sharing (VRS) scheme in which the prover has an input x and aims to secret-share x among the verifiers while proving that the input satisfies a certain predicate P i.e., $P(x) = 1$. The VRS scheme is stronger than the dCP because it ensures that all verifiers either output accept along with their respective shares if $P(x) =$ or they all output reject if the $P(x) \neq 1$ (unlike a dCP, where some verifiers can accept while others reject).

The ideal functionality for VRS, represented by $\mathcal{F}_{\mathsf{VRS}}$, is provided in Figure 2. The corresponding protocol Π_{VRS}, which is securely implemented $\mathcal{F}_{\mathsf{VRS}}$, is detailed in Figure 3. The formal theorem is given below.

Functionality $\mathcal{F}_{\mathsf{VRS}}$

The functionality $\mathcal{F}_{\mathsf{VRS}}$ communicates with a prover $\mathcal{P}$, a set of n verifiers $\mathcal{V} = \{\mathcal{V}_1, \ldots, \mathcal{V}_n\}$, and an adversary $\mathcal{A}$. It is parameterized by a verifier corruption-threshold t and predicate $P : \mathbb{F} \to \{0, 1\}$. Let $(\alpha_1, \ldots, \alpha_n)$ be publicly known elements in $\mathbb{F}$.

Share Phase. The share phase proceeds as follows.

1. The Prover $\mathcal{P}$ has input f which is a t-degree polynomial and sends the message $(\text{Input}, \mathsf{sid}, \mathcal{P}, f)$ to $\mathcal{F}_{\mathsf{VRS}}$. The verifiers $\mathcal{V}$ do not have any inputs.
2. Upon receiving the input from $\mathcal{P}$, $\mathcal{F}_{\mathsf{VRS}}$ checks if f is a t-degree polynomial and $P(f(0)) = 1$. Then, $\mathcal{F}_{\mathsf{VRS}}$ proceeds as follows:
   - If both checks pass, send $(\text{Output}, \mathsf{sid}, \mathcal{V}_i, f(\alpha_i), \mathsf{accept})$ to each verifier $\mathcal{V}_i$, who then outputs $(f(\alpha_i), \mathsf{accept})$, for all $i \in [n]$.
   - Otherwise, send $(\text{Output}, \mathsf{sid}, \mathcal{V}_i, \perp, \mathsf{reject})$ to all the verifiers and the verifiers output reject.

Figure 2: Ideal $\mathcal{F}_{\mathsf{VRS}}$ Functionality for Reed Solomon encoding

Theorem 4.3. *Let $t, n \in \mathbb{N}$ such that $t < n/3$ and P is a predicate. Then, the protocol Π_{VRS} between a prover $\mathcal{P}$ and n verifiers $\mathcal{V}_1, \ldots, \mathcal{V}_n$ described in Figure 3 securely realizes $\mathcal{F}_{\mathsf{VRS}}$ functionality in the $\mathcal{F}_{\mathsf{dCP}}$-hybrid model. The complexities are as follows:*

- *The prover's communication is $O(\mathbb{d} \log |\mathsf{C}| + \log^2 n)$ field elements for each verifier.*
- *The total communication complexity for all of the verifiers is*
  - *Offline phase: $n \cdot O(n^2 + \mathcal{BC}(n^2))$ field elements*
  - *Online phase: $n \cdot O(\mathbb{d} \log |\mathsf{C}| + \log^2 n + \mathcal{BC}(1))$ field elements*

Here, $\mathbb{d}$ represents the depth of the circuit C.

Efficiency. The complexities in the above theorem are for the case when the prover's input is a single field element. In this case, the prover sends to each of the verifiers the dCP proof, which costs $O(\mathbb{d} \log |\mathsf{C}| + \log^2 n)$ and an input share of size $O(1)$. The offline phase, run among the verifiers, involves n parallel invocations of VSS to secret-share one field element. We use the VSS scheme from [5], which has a communication cost of $O(n^2 + \mathcal{BC}(n^2))$ to secret-share a single field element among n parties. Thus, the total offline communication is n times the cost of a single VSS. In the online phase, each verifier receives a proof, which costs $O(\mathbb{d} \log |\mathsf{C}| + \log^2 n)$, and broadcasts a masked share, which costs $\mathcal{BC}(1)$.

We now extend to the case where the prover's input is a vector of d field elements. Similar to the efficiency for d input vector in dCP, the costs are computed with respect to a larger circuit $C_d : \mathbb{F}^{d \cdot (t+1)} \to \mathbb{F}^{d \cdot n}$. More precisely, the complexities are as follows[14]:

[14] Recall that the circuit C_d takes as input d t-degree polynomial and outputs evaluations of each of the d polynomials at n points.

Protocol Π_{VRS}

This protocol allows a prover $\mathcal{P}$ to verifiably secret share a value $x \in \mathbb{F}$ among n verifiers $\mathcal{V} = \{\mathcal{V}_1, \ldots, \mathcal{V}_n\}$ and prove that $P(x) = 1$. It is parameterized by a bound $n \geq 3t + 1$ where n is the number of verifiers, t is the number of corrupt verifiers and a predicate $P : \mathbb{F} \to \{0, 1\}$. Given predicate P, we can obtain a circuit $\mathsf{C} : \mathbb{F}^{t+1} \to \mathbb{F}^n$ and n relations $(\mathcal{R}_1, \ldots, \mathcal{R}_n)$ as per Definition 4.1. All the parties have access to an distributed commit-and-prove ideal functionality $\mathcal{F}_{\mathsf{dCP}}$, which is parameterized by $(\mathcal{R}_1, \ldots, \mathcal{R}_n)$. Let λ be the security parameter.

Input & Output. $\mathcal{P}$ has a t-degree polynomial f and the verifiers $\mathcal{V}$ have no inputs. If $P(f(0)) = 1$ holds, then each verifier $\mathcal{V}_i \in \mathcal{V}$ outputs $(f(\alpha_i), \mathsf{accept})$; Otherwise, all verifiers output reject.

Offline Phase. The parties interactively generate an RRS encoding of a random value r where $(\mathsf{rsh}_1, \ldots, \mathsf{rsh}_{n_s}) \leftarrow \mathsf{Enc}(r)$ and each verifier $\mathcal{V}_i$ receives shares rsh_i for all $i \in [n]$. Each verifier $\mathcal{V}_i \in \mathcal{V}$ proceeds as follows:

1. Sample a random value $r_i \in \mathbb{F}$ and secret-share it among the other verifiers using a VSS scheme such that $\mathcal{V}_j$ receives $\mathsf{rsh}_j^{(i)}$.
2. Upon the completion of all n VSS instances, $\mathcal{V}_i$ computes and ouptuts the random share $\mathsf{rsh}_i = \sum_{j \in [n]} \mathsf{rsh}_i^{(j)}$.

Sharing Phase.

1. **[Input Sharing]** Prover $\mathcal{P}$ with a secret x proceeds as follows.
   (a) Sample a t-degree polynomial $f(\cdot)$ such that $f(0) = x$ and compute $\mathsf{sh}_i = f(\alpha_i)$ for all $i \in [n]$. Then, invoke the Commit Phase of $\mathcal{F}_{\mathsf{dCP}}$ as the Prover with input $(\mathsf{Commit}, \mathsf{sid}, \mathcal{P}, f)$.
   (b) Invoke the Prove phase of $\mathcal{F}_{\mathsf{dCP}}$ as a Prover with input $(\text{dCP-prover}, \mathsf{sid}, \mathcal{P}, \mathsf{sh}_i, \mathcal{V}_i)$.
2. Upon receiving the message $(\text{dCP-Proof}, \mathsf{sid}, \mathsf{sh}_i, \mathsf{happy}_i)$ from $\mathcal{F}_{\mathsf{dCP}}$, if $\mathsf{happy}_i = \mathsf{accept}$, then broadcast $(\mathsf{valid\text{-}proof}, \mathcal{V}_i, \mathsf{msh}_i)$ where the masked share $\mathsf{msh}_i := \mathsf{sh}_i + \mathsf{rsh}_i$, otherwise set $\mathsf{msh}_i := \perp$ and broadcast nothing.
3. **[Consistency Check]** Let the broadcasted message from each verifier $\mathcal{V}_i \in \mathcal{V}$ be denoted by $(\mathsf{valid\text{-}proof}, \mathcal{V}_i, \mathsf{msh}_i')$. If the masked shares obtained from verifiers' broadcasts, denoted by $(\mathsf{msh}_1', \ldots, \mathsf{msh}_n')$, form a valid $\mathsf{RS}_{\mathbb{F},n,t+1}$ code with at most t errors (i.e., decoding succeeds on the masked shares).
4. **[Share Recovery]** Each verifier $\mathcal{V}_i$ *locally* computes its output as follows:
   (a) If the consistency check fails, then set $\mathsf{Share}_i := \perp$ and output $(\mathsf{reject}, \perp)$.
   (b) If the consistency check passes, then each verifier $\mathcal{V}_i \in \mathcal{V}$ outputs $(\mathsf{accept}, \mathsf{Share}_i)$ where Share_i is computed as follows:
      (i) **Keep Existing Share:** If $\mathsf{happy}_i = \mathsf{accept}$, then set $\mathsf{Share}_i := \mathsf{sh}_i$, or
      (ii) **Recover Share:** If $\mathsf{happy}_i = \mathsf{reject}$, then $\mathcal{V}_i$ needs to recover its share by computing $\mathsf{Share}_i := \mathsf{msh}_i'' - \mathsf{rsh}_i$ where $(\mathsf{msh}_1'', \ldots, \mathsf{msh}_n'')$ is obtained by error-correcting $(\mathsf{msh}_1', \ldots, \mathsf{msh}_n')$.

Figure 3: A VRS Protocol for predicate P

- The prover's communication is $O(\mathbb{d} \log |\mathsf{C}_d| + \log^2(d \cdot n))$ field elements for each verifier.
- The total communication complexity for all of the verifiers is
  - Offline phase: $n \cdot d \cdot O(n^2 + \mathcal{BC}(n^2))$ field elements
  - Online phase: $n \cdot O(\mathbb{d} \log |\mathsf{C}_d| + \log^2(d \cdot n) + \mathcal{BC}(d))$ field elements

4.3 Input-Certified Secure Aggregation From VRS

We present our input-certified secure aggregation protocol, which utilizes verifiable relation sharing (VRS) scheme as a building block. At a high level, the output party wants to aggregate the inputs $X_i \in \mathbb{F}^d$ of each client cU_i, while ensuring that only well-formed inputs are included in the aggregation. In our setting, well-formedness of the inputs is captured via a predicate $P(\cdot) : \mathbb{F}^d \to \{0,1\}$. We say that an input is well-formed if and only if $P(X_i) = 1$. Formally, the output party wants to compute $\Sigma_{\mathcal{U}_1} P(X_i) \cdot X_i$.

To achieve this, we require that each client secret-share their input among the servers. The servers can then aggregate the shares and reconstruct the aggregated shares towards the output party. To tolerate an adversary corrupting clients maliciously, we need to ensure that the clients verifiably secret share their input among the servers and also prove that their input are well-formed. The VRS scheme described in the previous section provides this exact functionality. Hence, each client can use the VRS scheme to secret-share and prove the well-formedness of their inputs to the verifiers. Essentially, the secure aggregation protocol involves three main steps: (1) Each client shares their input using the VRS scheme, (2) the servers aggregate the shares of all well-formed inputs, and (3) The aggregated shares are sent to the output party, who can then reconstruct the aggregate from the shares. A detailed description of the protocol Π_{Agg} is given in Figure 10 within Appendix L.

Theorem 4.4. *Let $\mathsf{n}_s, \mathsf{t}_s \in \mathbb{N}$ such that $\mathsf{t}_s < \mathsf{n}_s/3$ and $P : \mathbb{F}^d \to \{0,1\}$ be an arbitrary predicate. Let $\mathcal{F}_{\mathsf{Agg}}$ be the ideal functionality given in Figure 4. The protocol Π_{Agg}, as outlined in Figure 10, securely realizes $\mathcal{F}_{\mathsf{Agg}}$ in the $\mathcal{F}_{\mathsf{VRS}}$-hybrid model among n_c clients each holding input vectors of length d with elements in some finite field $\mathbb{F}$, n_s servers, and an output party $\mathcal{O}$, which is secure against a static, malicious rushing adversary that can maliciously corrupt an arbitrary number of clients, up to t_s servers and the output party and ensures guaranteed output delivery where κ is the security parameter. Additionally, a client is required to engage in only a single round of communication.*

The complexities are as follows:

- *The communication between a client and each server is $O(\mathbb{d} \log|\mathsf{C}| + \log^2 \mathsf{n}_s)$ field elements.*
- *The total amortized communication complexity of the n_s servers for a suffienctly large d is*
  - *Offline phase: $\mathsf{n}_c \cdot \mathsf{n}_s \cdot O(\mathsf{n}_s^2 \cdot (d/\mathsf{n}_s) + (d/\mathsf{n}_s) \cdot \mathsf{n}_s^3) = O(\mathsf{n}_c \cdot d \cdot \mathsf{n}_s^3)$ field elements.*
  - *Online phase: $\mathsf{n}_c \cdot \mathsf{n}_s \cdot d \cdot O(\mathbb{d} \log|\mathsf{C}| + \log^2 \mathsf{n}_s + \mathsf{n}_s)$ field elements.*

Here, $\mathbb{d}$ represents the depth of the circuit C.

Efficiency. The complexity for the Π_{Agg} are the cost of running n_c VRS instances, one per client, where the client has a d-dimensional vector as input. We highlight two optimizations. Firstly, when the input vector length d is large, broadcast extension protocols [36, 52] can be employed. Specifically, broadcasting a sufficiently long message comprising L field elements costs $O(\mathsf{n}_s \cdot L)$. Secondly, we improve the offline phase of the VRS protocol, which results in a reduction of the server's communication by n_s. Given that there are n_c invocation to VRS, the servers needs to generate random sharing of $\mathsf{n}_c \cdot d$ values in the offline phase of the vRS. Previously, each server shared a random value using a VSS, which were then aggregated to generate a single random sharing. To obtain $\mathsf{n}_c \cdot d$ random sharing, each server needs to invoke $\mathsf{n}_c \cdot d$ VSS instances in the offline phase. Instead of aggregating, we employ a randomness extraction technique [55, 34, 27] to extract $O(\mathsf{t}_s)$ random sharing from the n_s VSS invocations. This reduces the overall cost by a factor of n_s.

5 A Lower Bound for Wrapped DP DME

As mentioned in the Introduction, it is possible to design other protocols for DP DME by wrapping pre- and post-processing algorithms around certified secure aggregation. These alternative "wrapped protocols" could asymptotically improve on our bound on error. But in this section we argue this is impossible for a large class of wrapped protocol. Specifically, our lower bound applies to any $\Pi = (\mathsf{PRE}, P, \mathsf{POST})$ where

(a) the post-processing function POST is a linear transformation of the aggregated value y_{agg} and (b) any coordinate in its estimate of the population mean has bias dominated by its standard deviation. Repeated usage of DP DME (e.g. in gradient descent) will accumulate bias over time so constraint (b) is practical. And, as far as we are aware, all prior work that use a secure aggregation primitive for DP DME satisfy (a) [40, 2, 19].

We note that our lower bound takes place in a stochastic setting, where data is drawn independently from a probability distribution and the quantity to estimate is the mean of that distribution. Protocols that compute the empirical mean (like ours) can be used for the stochastic setting, so the lower bound applies.

Theorem 5.1. *Fix arbitrary $\hat{d} \in \mathbb{N}$. Suppose $\Pi = (\mathsf{PRE}, P, \mathsf{POST})$ is an (ε, δ)-DP protocol that wraps around certified secure aggregation Π_{Agg} and $\mathsf{POST}(y) := yQ$ for some public matrix $Q \in \mathbb{R}^{\hat{d} \times d}$. If $\Pi(\mathbf{D}^n)$ is an estimate of $\mathbb{E}[\mathbf{D}]$ whose standard deviation of error in any coordinate j exceeds the bias in j, then there is an attack K such that*

$$\mathbb{E}\left[\left\|\Pi^K(\mathbf{D}^n) - \mathbb{E}[\mathbf{D}]\right\|_2^2\right] \geq c \cdot \min\left(\frac{t^2}{n^2} \cdot \frac{d}{\varepsilon^2 n}, \frac{t^2}{n}\right).$$

where c is a universal constant.

Proof. We first recall a prior result in the DP literature: any differentially private algorithm that performs mean estimation must have $\approx d/\varepsilon^2 n^2$ expected squared error [42, 61].[15]

Theorem 5.2 (Adapted from [42, 61])**.** *For every $n, d \in \mathbb{N}$, $\varepsilon > 0$, and $\delta < 1/96n$, let $\mathcal{M}_{\varepsilon,\delta}$ denote the class of all (ε, δ)-private algorithms and let $\mathcal{D}$ denote the class of all product distributions on $\{\pm 1/\sqrt{d}\}^d$. There is a constant κ where*

$$\min_{M \in \mathcal{M}_{\varepsilon,\delta}} \max_{\mathbf{D} \in \mathcal{D}} \mathop{\mathbb{E}}_{X \sim \mathbf{D}^n, M}\left[\|M(X) - \mathbb{E}[\mathbf{D}]\|_2^2\right] \geq \kappa \cdot \min\left(\frac{d}{\varepsilon^2 n^2}, 1\right)$$

We use this theorem to prove, by way of a probabilistic argument, the existence of a strategy that introduces $\Omega(\frac{t^2}{n^2} \cdot \frac{d}{\varepsilon^2 n})$ squared bias. We defer details and analysis to Appendix F, but sketch the intuition here. If Π produces low-bias estimates when all clients are honest, the preceding theorem is effectively a lower bound on variance. Moreover, if POST is linear, each of the honest clients must contribute a $1/n$ fraction of that variance. This means the set of inputs an honest client can compute via PRE must be large enough to include an extreme value $\tilde{v}$.

Lemma 5.3. *Fix arbitrary $\hat{d} \in \mathbb{N}$. Suppose $\Pi = (\mathsf{PRE}, P, \mathsf{POST})$ is a protocol that wraps around certified secure aggregation Π_{Agg} and $\mathsf{POST}(y) := yQ$ for some public matrix $Q \in \mathbb{R}^{\hat{d} \times d}$. If $\Pi(\mathbf{D}^n)$ is an estimate of $\mathbb{E}[\mathbf{D}]$ such that the standard deviation of error in any coordinate j exceeds the bias in j, then there exists a vector $\tilde{v} \in \mathbb{F}^{\hat{d}}$ such that*

$$\left\|f(\tilde{v}) \cdot \tilde{v}Q - \mathop{\mathbb{E}}_{v \sim \mathsf{PRE}(\mathbf{D})}[P(v) \cdot vQ]\right\|_2^2 \geq \frac{\kappa}{2} \cdot \min\left(\frac{d}{\varepsilon^2 n^3}, 1/n\right)$$

where κ is the constant from Theorem 5.2.

Malicious clients can use that $\tilde{v}$ as input to certified secure aggregation instead of executing PRE on a sample from $\mathbf{D}$. □

To argue that our protocol matches this lower bound, we perform case analysis over d. In the regime where $d = O(\varepsilon^2 n)$, the bound simplifies to $\Omega(t^2/n^2)$. This is the same up to logarithmic factors as the first term in Theorem 3.6. In the regime where $d = \omega(\varepsilon^2 n)$ but $d = O(\varepsilon^2 n^2)$, the lower bound is dominated by a $t^2/n^2 \cdot d/\varepsilon^2 n$ term. Once again, this is the same as the first term in Theorem 3.6.

[15] We remark that lower bounds in this section assume that data is drawn i.i.d. and the output party desires a mean of the underlying distribution. Although our DBM only estimates the mean of the data, it can be used as a proxy for the distribution's mean, so our asymptotic bounds will match up to log factors.

References

[1] Masayuki Abe and Serge Fehr. Adaptively secure feldman VSS and applications to universally-composable threshold cryptography. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 317–334. Springer, 2004.

[2] Naman Agarwal, Peter Kairouz, and Ziyu Liu. The skellam mechanism for differentially private federated learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 5052–5064, 2021.

[3] Andris Ambainis, Markus Jakobsson, and Helger Lipmaa. Cryptographic randomized response techniques. In Feng Bao, Robert H. Deng, and Jianying Zhou, editors, *Public Key Cryptography - PKC 2004, 7th International Workshop on Theory and Practice in Public Key Cryptography, Singapore, March 1-4, 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 425–438. Springer, 2004.

[4] Benny Applebaum, Eliran Kachlon, and Arpita Patra. Verifiable relation sharing and multi-verifier zero-knowledge in two rounds: Trading nizks with honest majority - (extended abstract). In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 33–56. Springer, 2022.

[5] Michael Backes, Aniket Kate, and Arpita Patra. Computational verifiable secret sharing revisited. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 590–609. Springer, 2011.

[6] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 638–667. Springer, 2019.

[7] Laasya Bangalore, Mohammad Hossein Faghihi Sereshgi, Carmit Hazay, and Muthuramakrishnan Venkitasubramaniam. Flag: A framework for lightweight robust secure aggregation. In Joseph K. Liu, Yang Xiang, Surya Nepal, and Gene Tsudik, editors, *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security, ASIA CCS 2023, Melbourne, VIC, Australia, July 10-14, 2023*, pages 14–28. ACM, 2023.

[8] Soumya Basu, Alin Tomescu, Ittai Abraham, Dahlia Malkhi, Michael K. Reiter, and Emin Gün Sirer. Efficient verifiable secret sharing with share recovery in BFT protocols. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, pages 2387–2402. ACM, 2019.

[9] Amos Beimel, Kobbi Nissim, and Eran Omri. Distributed private data analysis: Simultaneously solving how and what. In David A. Wagner, editor, *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, volume 5157 of *Lecture Notes in Computer Science*, pages 451–468. Springer, 2008.

[10] James Bell, Adrià Gascón, Tancrède Lepoint, Baiyu Li, Sarah Meiklejohn, Mariana Raykova, and Cathie Yun. ACORN: input validation for secure aggregation. *IACR Cryptol. ePrint Arch.*, page 1461, 2022.

[11] James Henry Bell, Kallista A. Bonawitz, Adrià Gascón, Tancrède Lepoint, and Mariana Raykova. Secure single-server aggregation with (poly)logarithmic overhead. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 1253–1269. ACM, 2020.

[12] Jonas Böhler. *Input Secrecy & Output Privacy: Efficient Secure Computation of Differential Privacy Mechanisms*. PhD thesis, Karlsruhe Institute of Technology, Germany, 2021.

[13] Jonas Böhler and Florian Kerschbaum. Secure multi-party computation of differentially private median. In Srdjan Capkun and Franziska Roesner, editors, *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pages 2147–2164. USENIX Association, 2020.

[14] Kallista A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1175–1191. ACM, 2017.

[15] Lukas Burkhalter, Hidde Lycklama, Alexander Viand, Nicolas Küchler, and Anwar Hithnawi. Rofl: Attestable robustness for secure federated learning. *CoRR*, abs/2107.03311, 2021.

[16] Christian Cachin, Klaus Kursawe, Anna Lysyanskaya, and Reto Strobl. Asynchronous verifiable secret sharing and proactive cryptosystems. In Vijayalakshmi Atluri, editor, *Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS 2002, Washington, DC, USA, November 18-22, 2002*, pages 88–97. ACM, 2002.

[17] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Data poisoning attacks to local differential privacy protocols. In Michael Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 947–964. USENIX Association, 2021.

[18] TH Hubert Chan, Elaine Shi, and Dawn Song. Optimal lower bound for differentially private multi-party aggregation. In *European Symposium on Algorithms*, pages 277–288. Springer, 2012.

[19] Wei-Ning Chen, Ayfer Özgür, and Peter Kairouz. The poisson binomial mechanism for unbiased federated learning with secure aggregation. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 3490–3506. PMLR, 2022.

[20] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *CoRR*, abs/1712.05526, 2017.

[21] Albert Cheu, Matthew Joseph, Jieming Mao, and Binghui Peng. Shuffle private stochastic convex optimization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[22] Albert Cheu, Adam D. Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 375–403. Springer, 2019.

[23] Albert Cheu, Adam D. Smith, and Jonathan R. Ullman. Manipulation attacks in local differential privacy. In *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*, pages 883–900. IEEE, 2021.

[24] Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 383–395. IEEE Computer Society, 1985.

[25] Richard Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In Juris Hartmanis, editor, *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 364–369. ACM, 1986.

[26] Henry Corrigan-Gibbs and Dan Boneh. Prio: Private, robust, and scalable computation of aggregate statistics. In Aditya Akella and Jon Howell, editors, *14th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2017, Boston, MA, USA, March 27-29, 2017*, pages 259–282. USENIX Association, 2017.

[27] Ivan Damgård and Jesper Buus Nielsen. Scalable and unconditionally secure multiparty computation. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, volume 4622 of *Lecture Notes in Computer Science*, pages 572–590. Springer, 2007.

[28] Jinshuo Dong, Aaron Roth, and Weijie J. Su. Gaussian differential privacy. *CoRR*, abs/1905.02383, 2019.

[29] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, 2006.

[30] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006.

[31] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 2014.

[32] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2468–2479. SIAM, 2019.

[33] Paul Feldman. A practical scheme for non-interactive verifiable secret sharing. In *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 427–437. IEEE Computer Society, 1987.

[34] Paul Feldman and Silvio Micali. Optimal algorithms for byzantine agreement. In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 148–161. ACM, 1988.

[35] Matthias Fitzi, Juan A. Garay, Shyamnath Gollakota, C. Pandu Rangan, and K. Srinathan. Round-optimal and efficient verifiable secret sharing. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 329–342. Springer, 2006.

[36] Chaya Ganesh and Arpita Patra. Optimal extension protocols for byzantine broadcast and agreement. *Distributed Comput.*, 34(1):59–77, 2021.

[37] Rosario Gennaro, Yuval Ishai, Eyal Kushilevitz, and Tal Rabin. The round complexity of verifiable secret sharing and secure multicast. In Jeffrey Scott Vitter, Paul G. Spirakis, and Mihalis Yannakakis, editors, *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 580–589. ACM, 2001.

[38] Amir Herzberg, Markus Jakobsson, Stanislaw Jarecki, Hugo Krawczyk, and Moti Yung. Proactive public key and signature systems. In Richard Graveman, Philippe A. Janson, Clifford Neuman, and Li Gong, editors, *CCS '97, Proceedings of the 4th ACM Conference on Computer and Communications Security, Zurich, Switzerland, April 1-4, 1997*, pages 100–110. ACM, 1997.

[39] Amir Herzberg, Stanislaw Jarecki, Hugo Krawczyk, and Moti Yung. Proactive secret sharing or: How to cope with perpetual leakage. In Don Coppersmith, editor, *Advances in Cryptology - CRYPTO '95, 15th Annual International Cryptology Conference, Santa Barbara, California, USA, August 27-31, 1995, Proceedings*, volume 963 of *Lecture Notes in Computer Science*, pages 339–352. Springer, 1995.

[40] Peter Kairouz, Ziyu Liu, and Thomas Steinke. The distributed discrete gaussian mechanism for federated learning with secure aggregation. *CoRR*, abs/2102.06387, 2021.

[41] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. Secure multi-party differential privacy. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2008–2016, 2015.

[42] Gautam Kamath and Jonathan R. Ullman. A primer on private statistics. *CoRR*, abs/2005.00010, 2020.

[43] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. What can we learn privately? In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 531–540. IEEE Computer Society, 2008.

[44] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 177–194. Springer, 2010.

[45] Jonathan Katz, Chiu-Yuen Koo, and Ranjit Kumaresan. Improving the round complexity of VSS in point-to-point networks. *Inf. Comput.*, 207(8):889–899, 2009.

[46] Ranjit Kumaresan, Arpita Patra, and C. Pandu Rangan. The round complexity of verifiable secret sharing: The statistical case. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 431–447. Springer, 2010.

[47] Yurii Lyubarskii and Roman Vershynin. Uncertainty principles and vector quantization. *IEEE Trans. Inf. Theory*, 56(7):3491–3501, 2010.

[48] Yiping Ma, Jess Woods, Sebastian Angel, Antigoni Polychroniadou, and Tal Rabin. Flamingo: Multi-round single-server secure aggregation with applications to private federated learning. Cryptology ePrint Archive, Paper 2023/486, 2023. https://eprint.iacr.org/2023/486.

[49] Ilya Mironov. Rényi differential privacy. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 263–275. IEEE Computer Society, 2017.

[50] Ilya Mironov, Omkant Pandey, Omer Reingold, and Salil P. Vadhan. Computational differential privacy. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 126–142. Springer, 2009.

[51] Tal Moran and Moni Naor. Polling with physical envelopes: A rigorous analysis of a human-centric protocol. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 88–108. Springer, 2006.

[52] Kartik Nayak, Ling Ren, Elaine Shi, Nitin H. Vaidya, and Zhuolun Xiang. Improved extension protocols for byzantine broadcast and agreement. In Hagit Attiya, editor, *34th International Symposium on Distributed Computing, DISC 2020, October 12-16, 2020, Virtual Conference*, volume 179 of *LIPIcs*, pages 28:1–28:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[53] Thien Duc Nguyen, Phillip Rieger, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Ahmad-Reza Sadeghi, Thomas Schneider, and Shaza Zeitouni. FLGUARD: secure and private federated learning. *CoRR*, abs/2101.02281, 2021.

[54] Arpita Patra, Ashish Choudhary, Tal Rabin, and C. Pandu Rangan. The round complexity of verifiable secret sharing revisited. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 487–504. Springer, 2009.

[55] Arpita Patra, Ashish Choudhary, and C. Pandu Rangan. Simple and efficient asynchronous byzantine agreement with optimal resilience. In Srikanta Tirthapura and Lorenzo Alvisi, editors, *Proceedings of the 28th Annual ACM Symposium on Principles of Distributed Computing, PODC 2009, Calgary, Alberta, Canada, August 10-12, 2009*, pages 92–101. ACM, 2009.

[56] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

[57] Torben P. Pedersen. A threshold cryptosystem without a trusted party (extended abstract). In Donald W. Davies, editor, *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*, pages 522–526. Springer, 1991.

[58] Martin Pettai and Peeter Laud. Combining differential privacy and secure multiparty computation. In *Proceedings of the 31st Annual Computer Security Applications Conference, Los Angeles, CA, USA, December 7-11, 2015*, pages 421–430. ACM, 2015.

[59] Tal Rabin. Robust sharing of secrets when the dealer is honest or cheating. *J. ACM*, 41(6):1089–1109, 1994.

[60] Herbert Robbins. A remark on stirling's formula. *The American Mathematical Monthly*, 62:26–29, 1955.

[61] Thomas Steinke. *Upper and Lower Bounds for Privacy and Adaptivity in Algorithmic Data Analysis*. PhD thesis, Harvard University, Cambridge, MA, 2016.

[62] Timothy Stevens, Christian Skalka, Christelle Vincent, John Ring, Samuel Clark, and Joseph P. Near. Efficient differentially private secure aggregation for federated learning via hardness of learning with errors. In Kevin R. B. Butler and Kurt Thomas, editors, *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*, pages 1379–1395. USENIX Association, 2022.

[63] Kunal Talwar. Differential secrecy for distributed data and applications to robust differentially secure vector summation. *CoRR*, abs/2202.10618, 2022.

[64] Robin Vassantlal, Eduardo Alchieri, Bernardo Ferreira, and Alysson Bessani. COBRA: dynamic proactive secret sharing for confidential BFT services. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 1335–1353. IEEE, 2022.

[65] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[66] Larry Wasserman and Shuheng Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489):375–389, 2010.

[67] Yongji Wu, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Poisoning attacks to local differential privacy protocols for key-value data. In Kevin R. B. Butler and Kurt Thomas, editors, *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*, pages 519–536. USENIX Association, 2022.

[68] Jiaheng Zhang, Tiancheng Xie, Thang Hoang, Elaine Shi, and Yupeng Zhang. Polynomial commitment with a one-to-many prover and applications. In Kevin R. B. Butler and Kurt Thomas, editors, *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*, pages 2965–2982. USENIX Association, 2022.

A Preliminaries

A.1 Secure Aggregation

For any predicate $P : \mathbb{F}_q^d \to \{0,1\}$, let $\mathcal{F}_{\mathsf{Agg}}$ be the ideal functionality that takes a finite sequence of $\mathbb{Z}_q^d$ values $X_1, X_2, \ldots, X_{\mathsf{n_c}}$ as input and reports the sum of the values on which the predicate evaluates to 1. Formally, it outputs $\mathsf{Y}_{agg} = \sum_i P(X_i) \cdot X_i$. The ideal functionality is given in Figure 4.

Functionality $\mathcal{F}_{\mathsf{Agg}}$

The functionality $\mathcal{F}_{\mathsf{Agg}}$ communicates with the set of clients $\mathcal{U} = \{u_1, \ldots, u_{\mathsf{n_c}}\}$, an output party $\mathcal{O}$ and an adversary $\mathcal{A}$. It is parameterized by $P : \mathbb{F}_q^d \to \{0,1\}$, $\mathsf{n_c}$ and d, where P is the predicate used for certify the inputs, $\mathsf{n_c}$ is the number of clients and d denotes the size of client's input vector.

1. Upon receiving input ("Input", sid, u_i, X_i) from some new client $u_i \in \mathcal{U}$ where $X_i \in \mathbb{F}^d$, then store the client's input X_i.
2. Upon receiving ("Output", $\mathcal{O}$) from the output party $\mathcal{O}$, proceed as follows:
   - Compute the aggregate $\mathsf{Y}_{agg} = \sum_{u_i \in \mathcal{U}} P(X_i) \cdot X_i$ (note that this is equivalent to aggregating only inputs of clients that satisfy the predicate P).
   - Send ("Output", $\mathsf{sid}, \mathsf{Y}_{agg}$) to the output party $\mathcal{O}$ and halt.

Figure 4: Ideal Functionality for Secure Aggregation with Input Certification

A.2 Differential Privacy

We say that two distributions $\mathbf{P}, \mathbf{Q}$ are (ε, δ)-close—shorthanded as $\mathbf{P} \approx_{\varepsilon,\delta} \mathbf{Q}$—if, for all events E,

$$\mathbb{P}_{\eta \sim \mathbf{P}}[\eta \in E] \leq e^{\varepsilon} \cdot \mathbb{P}_{\eta \sim \mathbf{Q}}[\eta \in E] + \delta$$
$$\mathbb{P}_{\eta \sim \mathbf{Q}}[\eta \in E] \leq e^{\varepsilon} \cdot \mathbb{P}_{\eta \sim \mathbf{P}}[\eta \in E] + \delta$$

Definition A.1 (Approximate DP [30, 29])**.** A randomized algorithm M satisfies (ε, δ)-*approximate differential privacy* if, for any two datasets X, X' that differ on one client's data, $M(X) \approx_{\varepsilon,\delta} M(X')$.

Note that small parameter values imply a stronger privacy guarantee: at the limiting value of 0, the algorithm's output is completely independent of the input.

Let $\{\mathbf{P}_\lambda\}, \{\mathbf{Q}_\lambda\}$ be two sequences of distributions. We say that the sequences are $(\varepsilon, \delta = \text{negl}(\lambda))$-computationally close—shorthanded as $\{\mathbf{P}_\lambda\} \approx^c_{\varepsilon,\delta} \{\mathbf{Q}_\lambda\}$—if, for any PPT distinguisher A, every advice string z_λ of length $\text{poly}(\lambda)$, and sufficiently large λ,

$$\mathop{\mathbb{P}}_{\eta\sim\mathbf{P}_\lambda}\left[A(1^\lambda, z_\lambda, \eta) = 1\right] \leq e^\varepsilon \cdot \mathop{\mathbb{P}}_{\eta\sim\mathbf{Q}_\lambda}\left[A(1^\lambda, z_\lambda, \eta) = 1\right] + \delta$$
$$\mathop{\mathbb{P}}_{\eta\sim\mathbf{Q}_\lambda}\left[A(1^\lambda, z_\lambda, \eta) = 1\right] \leq e^\varepsilon \cdot \mathop{\mathbb{P}}_{\eta\sim\mathbf{P}_\lambda}\left[A(1^\lambda, z_\lambda, \eta) = 1\right] + \delta$$

Definition A.2 (Computational DP [50])**.** A randomized algorithm M parameterized by security parameter λ satisfies (ε, δ)-IND-CDP (also referred to as (ε, δ)-approximate computational differential privacy) if, for any two sequences of datasets $\{X_\lambda\}, \{X'_\lambda\}$ where $|X_\lambda| = |X'_\lambda| = \text{poly}(\lambda)$ and X_λ, X'_λ differ on on one client's data, $\{M_\lambda(X_\lambda)\} \approx^c_{\varepsilon,\delta} \{M_\lambda(X'_\lambda)\}$

A.3 Distributed Mean Estimation

Inspired by earlier work by Cheu, Smith, and Ullman [23], we define distributed mean estimation in the context of a generic manipulation game played between protocols and adversaries attempting to skew the outcome.

The Manipulation Game

Elements: $\mathsf{n_c}$ clients, n_s servers, an output party $\mathcal{O}$, a protocol Π they are meant to run, and an attack K.
Game Parameters: Number of corrupted clients $\mathsf{t_c}$ and corrupted servers $\mathsf{t_s}$.

1. The environment gives each client u_i some data X_i
2. The adversary chooses $\mathsf{t_c}$ clients (denoted $\mathcal{C}$) and $\mathsf{t_s}$ servers to corrupt
3. The honest clients (denoted $\mathcal{H}$) and honest servers follow the protocol, while the adversary compels the rest to launch attack K
4. $\mathcal{O}$ receives (or computes) the output $\Pi^K(X)$

Figure 5: A game played between a protocol and an adversary who wishes to reduce accuracy

In the case of DME, the environment chooses data X_i from $\mathcal{B}^d$, the Euclidean unit ball in d dimensions. We say that a protocol Π *solves distributed mean estimation (DME) with mean-squared error* α if, for any choice of input X and attack K, the manipulation game results in output $\Pi^K(X)$ that satisfies $\mathop{\mathbb{E}}_{\Pi,K}\left[(\Pi^K(X) - \frac{1}{\mathsf{n_c}}\sum_{i\in[\mathsf{n_c}]} X_i)^2\right] \leq \alpha$.

We will perform DME by wrapping pre- and post-processing functions around input-certified secure aggregation. The pre-processing injects noise that will ensure differential privacy.

B Related Work

Differentially Private Distributed Mean Estimation (DP-DME) and Secure Aggregation. As previously discussed, Kairouz et al. and Agarwal et al provide wrappings around secure aggregation that ensure DP. Unlike our work, they do not consider malicious clients and they do not add binomial noise to client data. The Poisson-Binomial mechanism (PBM) by Chen, Özgür, and Kairouz also differs with ours in those

Variable	Meaning
n	number of clients
t	number of malicious clients $0 \le t < n$
h	number of honest clients
d	dimensionality
$\mathcal{B}^d$	unit ball in d dimensions
$\Pi = (\mathsf{PRE}, P, \mathsf{POST})$	protocol that wraps around $\mathcal{F}_{\mathsf{Agg}}$
$\mathcal{F}_{\mathsf{Agg}}$	Ideal function which computes $\sum P(Y_i) \cdot Y_i$ on input $Y_1, Y_2, \ldots$
$K = (\mathcal{C}, \mathbf{T})$	attack where $\mathcal{C}$ is set of malicious clients and $\mathbf{T}$ determines strategy
$\mathcal{H}$	honest clients
$X_i[j]$	j-th element of row i (client i's data)
η	sample of noise
$\alpha, \varepsilon, \delta$	privacy parameters
μ, σ^2	mean, variance
g, b, τ, r	parameters of DBM (our protocol)
m	parameter of binomial mechanism
κ, S	constants
c, ℓ, θ	parameters of PBM (Chen et al. [19])

Table 2: Table of variables used in our DP DME (Sections 3 & 5, Appendix G).

Variable	Meaning
$\mathsf{n_c}$	number of clients
$\mathsf{t_c}$	number of malicious clients $0 \le \mathsf{t_c} < \mathsf{n_c}$
n_s	number of servers
$\mathsf{t_s}$	number of malicious servers
$\mathsf{d_s}$	number of dropped out servers
d	dimensionality
ℓ_s	packing parameter for packed secret sharing
$\mathsf{m_p}$	Number of blocks
β	number of columns per server in Share
$X_1, \ldots, X_{\mathsf{n_c}}$	Client's inputs
Y_{agg}	Output of secure aggregation
λ	Security Parameter

Table 3: Table of variables used in our Input Certified Secure Aggregation (Section 4)

respects [19]. But unlike Kairouz et al. and Agarwal et al., the PBM's clients feed *finite* values into the secure aggregator. This makes it possible to choose a predicate P that culls malicious values without disturbing the existing privacy analysis; we do so in Appendix G. There, we find large constants and extra logarithmic factors in the error bound are challenging to avoid simultaneously. In Figure 8, we compare the error bounds of our protocol with those of the PBM. Chen et al. argue the PBM satisfies Rényi DP (RDP) so to ensure an appropriate comparison, we employ a conversion from RDP to approx DP by Mironov [49]. The conversion can conceivably be improved but technical hurdles stand in our way. Appendix G.2.2 contains more information.

The three prior works discussed above ensure Rényi DP (RDP) and concentrated DP (CDP). Approximate DP, the guarantee of our protocol, does not have a counterpart to the state-of-the-art composition results for RDP and CDP. So on the face of it, our approximate DP protocol appears deficient. But we argue it is still competitive.

- Because each execution of our protocol is analyzed as a simulation of the Gaussian mechanism, we can partially leverage composition theorems that apply to the Gaussian mechanism. Roughly speaking, we can increase fidelity of Gaussian noise simulation without impacting the output party's estimate. The upshot is that approximate DP composition only determines the granularity of discretization while Gaussian-based composition determines error. We give more details in Section E.0.1.

- If we were to upgrade the standard secure aggregator in Kairouz et al.'s protocol with an input-certified one, we will be forced to use approximate DP anyway. This is because each client places nonzero probability on every integer, so any nontrivial predicate P would erase a client's contribution with nonzero probability δ. The same holds for the protocol by Agarwarl et al. Similar to our protocol, we can factor out the accumulation of δ from the consumption of privacy budget.

Protocols that wrap around secure aggregation resemble protocols in the shuffle model of DP. These are wrappings around a primitive that permutes values uniformly at random (see e.g. [22, 32, 6, 21]). Cheu, Joseph, Mao, and Peng describe a shuffle protocol for DP DME that uses binomial noise like we do [21]. There, the authors treat each of the d dimensions as a separate private protocol and employ an advanced composition theorem to bound the overall privacy parameter. We instead prove that it is possible to transfer privacy analysis of d-dimensional Gaussian noise to d-dimensional binomial noise. Our method results in an error bound that avoids a $\log d$ factor that appears in Cheu et al.'s work.

Talwar presents a client-server protocol for distributed mean estimation with input certification [63]. Security is ensured against semi-honest servers. Clients prove their contributions belong to a Euclidean ball with *differentially zero knowledge* proof: changing inputs to the protocol causes a change in the transcript distribution that is bounded in terms of the divergence that appears in the approximate differential privacy definition (see Definition A.1). But this is the sole appearance of differential privacy in the explicit construction, as the output party observes the aggregation of valid data without any noise. Furthermore, the presented protocol assumes parties can communicate real-valued numbers. Talwar does briefly discuss DP wrappings and sketches an argument that the protocol can be accurately approximated with finite precision, but we emphasize that our work treats finite precision and differential privacy as first-order concerns.

Several works combine secure aggregation with differential privacy. Keeler et al. (2023) introduces DPrio, a system that augments the well-known Prio system with differential privacy, achieving high utility comparable to centralized differential privacy. Like Prio, DPrio's threat model tolerates malicious corruption of clients and semi-honest corruption of all but one server. Additionally, it assumes the adversary can control the noise generated by only a small, constant number of clients. In contrast, our work is designed to tolerate malicious corruption of both clients and a fraction of the servers, and tolerates the adversary corrupting a larger number of clients.

In contrast, the work by Stevens et al. [62] tolerates a malicious adversary and operates within a model with a single central server, similar to classic secure aggregation protocols like those by Bonawitz et al. and Bell et al. Our approach differs by relying on an honest majority in the number of servers which gives us guaranteed output delivery. Additionally, in Stevens et al.'s model, clients are required to participate in multiple rounds even for a single iteration of secure aggregation, whereas our protocol aims to minimize client participation to just one round.

Beyond secure aggregation, other works such as [13, 12] study differentially private median protocol. Also, the works of [41, 58] explore combining general multiparty computation with differential privacy.

Manipulation Attacks against DP Protocols. Though studied on a per-protocol basis in earlier work, the concept of *manipulation attacks against DP protocols* was formalized by Cheu, Smith, and Ullman [23]. In these attacks, adversaries seek to distort the result of a computation (e.g. introduce bias to a mean estimate). Our DP protocol upper bounds the success of such attacks. We take advantage of multi-party computation while Cheu et al. restrict themselves to the local model, where the transcript of all plaintext messages from any client must satisfy DP. There, the skew of an estimate can worsen with smaller privacy budget ε and larger dimension d. A similar phenomenon can be found in our lower bound, though it is tempered by an $1/n_c$

factor. Another point of difference is that we focus on data in the ℓ_2 unit ball, while the prior work's primary results concern the ℓ_1 unit ball.

Verifiable Secret/Relation Sharing (VSS/VRS). Verifiable Secret Sharing (VSS) was initially introduced by Chor et al. [24] and has remained a significant research area in cryptography over the past two decades [39, 45, 46, 54, 59, 56, 57, 37, 35, 33, 16, 1]. However, these schemes often require the dealer to perform more work compared to the non-verifiable variant of secret-sharing. When it comes to verifiable secret-sharing among n parties, the dealer's efficiency may be hindered due to several factors.

Firstly, the dealer needs to engage in multiple rounds of interaction with the parties, some of which are broadcast rounds. In classical VSS schemes, the dealer needs to participate in multiple rounds to resolve any potential conflicts that may arise among the parties, some of the rounds may involve a broadcast. This requirement eliminates the straightforward use of many existing VSS schemes, as we aim to ensure that the dealer only needs to engage in one point-to-point round with the parties.

Second, the dealer may incur communication complexity quadratic in n, typically due to using bivariate polynomials [45, 54, 5].

Third, some schemes require a trusted setup and are computationally expensive [44]. These aspects pose limitations, particularly when considering the unreliability of clients who may drop out at any point during the execution.

Our techniques are closely related to the share recovery mechanisms that appear in the context of Proactive Secret Sharing (PSS) [38, 39] and Asynchronous VSS [16]. Both PSS and BFT-SMR involve parties that possess shares of some private value, with share recovery processes triggered if any party's share is missing. Unlike VSS, a notable advantage is that the dealer is not involved in the recovery process. However, a drawback is that the dealer may need to share additional polynomials to enable share recovery [8, 64] and a separate random polynomial may need to be generated for recovery of each share. But these techniques extend beyond the synchronous setting and parties can trigger share recovery at any point of time. In contrast, our focus is on the synchronous setting and share recovery may be triggered only at the end of the first round (after the dealer shares the secret among the parties). This simplifies the problem and it is sufficient to use any existing VSS to secret-share the random recovery polynomials among the parties.

Lastly, our interest extends beyond verifiably secret sharing data to also proving that the secret-shared data satisfies certain properties. This aligns with the Verifiable Relation Sharing (VRS) functionality introduced by Applebaum et al. [4]. However, their VRS construction is hindered by similar way to classical VSS as mentioned above, as it incurs multiple rounds dealer participation, high communication complexity for the dealer (due to the use of bivariate polynomials), and the need for a trusted setup.

C De Moivre-Laplace Theorem for finite variance

Theorem 3.4 (Finite-value De Moivre-Laplace)**.** *Fix any $\varepsilon \in (0,1)$, $\delta \in (0, 2e^{-6})$ and any even $m > \frac{12}{\varepsilon^2}\ln^2\frac{2}{\delta}$.*

$$\mathsf{round}(\mathbf{N}(0, \sigma^2 = m/4)) \approx_{\varepsilon,\delta} \mathbf{Bin}(m, 1/2) - m/2$$

Proof. Our goal is to show, for any event E, that

$$\mathbb{P}[\mathsf{round}(\mathcal{N}(0, m/4)) \in E] \le e^{\varepsilon} \cdot \mathbb{P}[\mathbf{Bin}(m, 1/2) - m/2 \in E] + \delta \tag{1}$$

$$\mathbb{P}[\mathbf{Bin}(m, 1/2) - m/2 \in E] \le e^{\varepsilon} \cdot \mathbb{P}[\mathsf{round}(\mathcal{N}(0, m/4)) \in E] + \delta \tag{2}$$

We prove (1) first. Define $T_0 := \left[-\sqrt{\frac{m}{2}\ln\frac{2}{\delta}},\ +\sqrt{\frac{m}{2}\ln\frac{2}{\delta}}\right]$. By using a Chernoff bound, we have the concentration inequality $\mathbb{P}[\mathcal{N}(0, m/4) \notin T_0] \le \delta$. The concentration inequality for the rounded Gaussian must concern a wider interval, as rounding could move mass out from T_0. It can be easily shown that for $\delta < 2e^{-6}$ and $m > 10\ln\frac{2}{\delta}$, $\sqrt{\frac{3m}{5}\ln\frac{2}{\delta}} > 1 + \sqrt{\frac{m}{2}\ln\frac{2}{\delta}}$ so $T = \left[-\sqrt{\frac{3m}{5}\ln\frac{2}{\delta}},\ +\sqrt{\frac{3m}{5}\ln\frac{2}{\delta}} + 1\right]$ is big enough to contain the rounded-up values.

Now we express the left-hand side of (1) in terms of T:

$$\begin{aligned}
&\mathbb{P}[\mathsf{round}(\mathcal{N}(0,m/4)) \in E] \\
&\leq \mathbb{P}[\mathsf{round}(\mathcal{N}(0,m/4)) \in E \cap T] + \delta \\
&= \frac{\mathbb{P}[\mathsf{round}(\mathcal{N}(0,m/4)) \in E \cap T]}{\mathbb{P}[\mathbf{Bin}(m,1/2) - h/2 \in E \cap T]} \cdot \mathbb{P}[\mathbf{Bin}(m,1/2) - m/2 \in E \cap T] + \delta
\end{aligned}$$

So it will suffice to upper-bound the ratio by $\exp(\varepsilon')$.

Our first step is to find an expression for the PMF of the rounded Gaussian distribution. By definition of rounding, the mass placed at distance s from the mean is the change in the Gaussian CDF from $s-0.5$ to $s+0.5$:

$$\mathbb{P}[\mathsf{round}(\mathcal{N}(0,h/4)) = s] = \frac{1}{2}\left[\mathrm{erf}\left(\frac{s+1/2}{\sigma\sqrt{2}}\right) - \mathrm{erf}\left(\frac{s-1/2}{\sigma\sqrt{2}}\right)\right] \tag{3}$$

If (u_0, v_0) is a point on $\mathrm{erf}(u)$, the tangent line at (u_0, v_0) has equation $v - v_0 = \frac{d\,\mathrm{erf}}{du}(u_0) \cdot (u - u_0) = \frac{2}{\sqrt{\pi}}\exp(-u_0^2) \cdot (u - u_0)$. Given a second point (u_1, v_1) on $\mathrm{erf}(u)$, we have

$$\begin{aligned}
v_1 - v_0 &= \frac{2}{\sqrt{\pi}}\exp(-u_0^2) \cdot (u - u_0) - \frac{2}{\sqrt{\pi}}\exp(-u_1^2) \cdot (u - u_1) \\
&= \frac{2}{\sqrt{\pi}}\exp(-u_0^2) \cdot (u - u_0) + \frac{2}{\sqrt{\pi}}\exp(-u_1^2) \cdot (u_1 - u) \\
&= \frac{2}{\sqrt{\pi}} \cdot \frac{u_1 - u_0}{2} \cdot \left(\exp(-u_0^2) + \exp(-u_1^2)\right) && \text{(for } u = (u_0+u_1)/2\text{)}
\end{aligned}$$

Thus,

$$\begin{aligned}
(3) &= \frac{1}{2} \cdot \left[\frac{1}{\sigma\sqrt{2\pi}} \cdot \left(\exp\left(-\frac{s^2 - s + 1/4}{2\sigma^2}\right) + \exp\left(-\frac{s^2 + s + 1/4}{2\sigma^2}\right)\right)\right] \\
&= \frac{1}{\sqrt{2\pi h}} \cdot \left(\exp\left(-\frac{s^2 - s + 1/4}{h/2}\right) + \exp\left(-\frac{s^2 + s + 1/4}{h/2}\right)\right) \\
&\in \left[\frac{\sqrt{2}}{\sqrt{\pi h}} \cdot \exp\left(\frac{-2s^2 - 2s - 1/2}{h}\right),\ \frac{\sqrt{2}}{\sqrt{\pi m}} \cdot \exp\left(\frac{-2s^2 + 2s - 1/2}{h}\right)\right]
\end{aligned} \tag{4}$$

Now we find an expression for the binomial's mass function: for any integer $s \in \left[0, \sqrt{\frac{m}{2}\ln\frac{2}{\delta}} + 1\right]$,[16]

$$\begin{aligned}
&\mathbb{P}\left[\mathbf{Bin}(m,1/2) = \underbrace{m/2 + s}_{k}\right] \\
&\geq \sqrt{\frac{m}{2\pi k(m-k)}} \cdot \left(\frac{m}{2k}\right)^k \left(\frac{m}{2(m-k)}\right)^{h-k} \cdot \exp\left(-\frac{m}{12k(m-k)}\right) && \text{(from [60])} \\
&= \sqrt{\frac{m}{2\pi k(m-k)}} \cdot \exp\left(-k\ln\frac{2k}{m} + (k-m)\ln\frac{2(m-k)}{m} - \frac{m}{12k(h-k)}\right) \\
&= \sqrt{\frac{1}{2\pi(1/2 + s/m)(m/2 - s)}} \cdot \exp\left(-k\ln\frac{2k}{m} + (k-m)\ln\frac{2(m-k)}{m} - \frac{1}{12(1/2 + s/m)(m/2 - s)}\right) \\
&= \sqrt{\frac{1}{2\pi(m/4 - s^2/m)}} \cdot \exp\left(-k\ln\left(1 + \frac{2s}{m}\right) + (k-m)\ln\left(1 - \frac{2s}{m}\right) - \frac{1}{12(m/4 - s^2/m)}\right)
\end{aligned} \tag{5}$$

[16] The distribution of interest is symmetric about $m/2$, so upper and lower bounds for the mass at $m/2 + s$ also hold for $h/2 - s$.

We focus on the two terms involving logarithms. We use the identity $\ln(1+z) = z - z^2/2 + z^3/3 - \ldots$:

$$\begin{aligned}
&-k\ln\left(1+\frac{2s}{m}\right)+(k-m)\ln\left(1-\frac{2s}{m}\right)\\
&=-k\cdot\left(\frac{2s}{m}-\frac{1}{2}\left(\frac{2s}{m}\right)^2+\ldots\right)+(k-m)\cdot\left(-\frac{2s}{m}-\frac{1}{2}\left(-\frac{2s}{m}\right)^2+\frac{1}{3}\cdot\left(-\frac{2s}{m}\right)^3\ldots\right)\\
&=\left(-\frac{m}{2}-s\right)\cdot\left(\frac{2s}{m}-\frac{1}{2}\left(\frac{2s}{m}\right)^2+\ldots\right)-\left(\frac{m}{2}-s\right)\cdot\left(-\frac{2s}{m}-\frac{1}{2}\left(-\frac{2s}{m}\right)^2+\frac{1}{3}\cdot\left(-\frac{2s}{m}\right)^3\ldots\right)\\
&=-\frac{2s^2}{m}+s\sum_{i=1}^{\infty}\left(\frac{2s}{m}\right)^{2i+1}\left(\frac{1}{i+1}-\frac{1}{2i+1}\right)\\
&=-\frac{2s^2}{m}+s\left(-\frac{m}{2s}\ln\left(1-\frac{4s^2}{(m)^2}\right)-\tanh^{-1}\frac{2s}{h}\right)\\
&\in\left[-\frac{2s^2}{m},\ -\frac{2s^2}{m}+\frac{s}{2}\left(\frac{2s}{m}\right)^3\right] \qquad (6)
\end{aligned}$$

The lower bound comes from the fact that each term in the series is positive. The upper bound is derived from the fact that $2s/m \le \sqrt{\frac{12}{5m}\ln\frac{2}{\delta}} < 4/5$ for all $m > 10\ln 2/\delta$.

So by combining (4), (5), and (6), we finally have

$$\begin{aligned}
&\frac{\mathbb{P}[\mathsf{round}(\mathcal{N}(0,m/4))\in E\cap T]}{\mathbb{P}[\mathbf{Bin}(m,1/2)-h/2\in E\cap T]}\\
&\le 2\sqrt{1/4-s^2/h^2}\cdot\exp\left(\frac{-2s^2+2s-1/2}{m}-\left(-\frac{2s^2}{m}-\frac{1}{12(m/4-s^2/m)}\right)\right)\\
&=2\sqrt{1/4-s^2/m^2}\cdot\exp\left(\frac{2s-1/2}{m}+\frac{1}{3m-12s^2/m}\right)\\
&\le\exp\left(\sqrt{\frac{12}{5m}\ln\frac{2}{\delta}}+\frac{1}{3m-36\ln(2/\delta)/5}\right)
\end{aligned}$$

If $m > \frac{5}{\varepsilon^2}\ln\frac{2}{\delta}$, then

$$\begin{aligned}
&\exp\left(\sqrt{\frac{12}{5m}\ln\frac{2}{\delta}}+\frac{5}{15m-36\ln(2/\delta)}\right)\\
&\le\varepsilon\sqrt{\frac{12}{25}}+\frac{5\varepsilon^2}{75\ln(2/\delta)-36\varepsilon^2\ln(2/\delta)}\\
&<\varepsilon\cdot\left(\sqrt{\frac{12}{25}}+\frac{5}{39\ln(2/\delta)}\right) \qquad (\varepsilon<1)\\
&<\varepsilon
\end{aligned}$$

It remains to prove (2). We use very similar steps as in the proof of (1). Hoeffding's inequality implies the binomial random variable lies outside of T with probability $\le\delta$. Then we set out to upper bound the ratio

$$\frac{\mathbb{P}[\mathbf{Bin}(m,1/2)-m/2\in E\cap T]}{\mathbb{P}[\mathsf{round}(\mathcal{N}(0,m/4))\in E\cap T]}$$

We previously lower bounded the binomial mass function, so now we upper bound that function. We repeat

the analysis from before but apply upper bounds to the numerator instead of the denominator:

$$\begin{aligned}
&\mathbb{P}\left[\mathbf{Bin}(m,1/2) = \underbrace{m/2+s}_{k}\right] \\
&\leq \sqrt{\frac{1}{2\pi(1/2+s/m)(m/2-s)}} \cdot \exp\left(-k\ln\left(1+\frac{2s}{h}\right)+(k-h)\ln\left(1-\frac{2s}{h}\right)+\frac{1}{12h}\right) && \text{(from [60])} \\
&\leq \sqrt{\frac{1}{2\pi(1/2+s/m)(m/2-s)}} \cdot \exp\left(-\frac{2s^2}{m}+\frac{s}{2}\left(\frac{2s}{m}\right)^3+\frac{1}{12m}\right) && \text{(via (6))}
\end{aligned}$$

and therefore, by way of (4),

$$\begin{aligned}
&\frac{\mathbb{P}[\mathbf{Bin}(m,1/2)\in E\cap T]}{\mathbb{P}[\mathsf{round}(\mathcal{N}(m/2,m/4))\in E\cap T]} \\
&\leq \frac{1}{2}\sqrt{\frac{1}{(1/2+s/m)(1/2-s/m)}} \cdot \exp\left(-\frac{2s^2}{m}+\frac{4s^4}{m^3}+\frac{1}{12m}+\frac{2s^2+2s+1/2}{m}\right) \\
&= \sqrt{\frac{1}{1-4s^2/m^2}} \cdot \exp\left(\frac{4s^4}{m^3}+\frac{7}{12m}+\frac{2s}{m}\right)
\end{aligned}$$

Now we substitute $s \leq \sqrt{\frac{3m}{5}\ln\frac{2}{\delta}}$:

$$\begin{aligned}
&\leq \sqrt{\frac{1}{1-12\ln(2/\delta)/5m}} \cdot \exp\left(\frac{36\ln^2(2/\delta)}{25m}+\frac{7}{12h}+\sqrt{\frac{12}{5m}\ln\frac{2}{\delta}}\right) \\
&= \exp\left(\frac{1}{2}\ln\frac{5m}{5m-12\ln(2/\delta)}+\frac{36\ln^2(2/\delta)}{25h}+\frac{7}{12m}+\sqrt{\frac{12}{5m}\ln\frac{2}{\delta}}\right) \\
&\leq \exp\left(\frac{6\ln(2/\delta)}{5m-12\ln(2/\delta)}+\frac{36\ln^2(2/\delta)}{25m}+\frac{7}{12m}+\sqrt{\frac{12}{5m}\ln\frac{2}{\delta}}\right)
\end{aligned}$$

If $m > \frac{12}{\varepsilon^2}\ln^2(2/\delta)$, then

$$\begin{aligned}
&\frac{6\ln(2/\delta)}{5m-12\ln(2/\delta)}+\frac{36\ln^2(2/\delta)}{25m}+\frac{7}{12m}+\sqrt{\frac{12}{5m}\ln\frac{2}{\delta}} \\
&< \frac{6\varepsilon^2\ln(2/\delta)}{60\ln^2(2/\delta)-12\varepsilon^2\ln(2/\delta)}+\frac{3}{25}\varepsilon^2+\frac{7}{144}\varepsilon^2+\varepsilon\sqrt{\frac{1}{5}} \\
&< \frac{6\varepsilon^2\ln(2/\delta)}{48\varepsilon\ln(2/\delta)}+\frac{3}{25}\varepsilon^2+\frac{7}{144}\varepsilon^2+\varepsilon\sqrt{\frac{1}{5}} \\
&< \varepsilon\cdot\left(\frac{1}{8}+\frac{3}{25}+\frac{7}{144}+\sqrt{\frac{1}{5}}\right) < \varepsilon
\end{aligned}$$

□

D Proofs for Binomial Mechanism and DBM

We now present proofs deferred from Sections 3.1 and 3.2

Theorem 3.5 (Privacy of BM). *Let f be any Δ-sensitive function over the integers $\mathbb{Z}^d$, let W, W' be matrices differing on one row. Fix any ε_0, ε_1, δ_0, and δ_1 in the interval $(0,1)$. If m is an even integer such that $m >$*

$\max\left(\frac{12}{\varepsilon_0^2}\ln^2\frac{2}{\delta_0},\ \frac{8\Delta^2}{\varepsilon_1^2}\ln\frac{5}{4\delta_1}\right)$, *then for any event* E,

$$\mathbb{P}\Big[\mathsf{BM}_{f,m}(W)\in E\Big]\leq \exp(2d\varepsilon_0+\varepsilon_1)\cdot\mathbb{P}\Big[\mathsf{BM}_{f,m}(W')\in E\Big] + 2\exp(d\varepsilon_0+\varepsilon_1)\cdot d\delta_0+\exp(d\varepsilon_0)\cdot\delta_1$$

Proof. We bound the left-hand probability by a probability involving GM.

$$\begin{aligned}
&\mathbb{P}\Big[\mathsf{BM}_{f,m}(W)\in E\Big]\\
&=\mathop{\mathbb{P}}_{\eta[j]\sim\mathbf{Bin}(m,1/2)-m/2}\Big[\eta\in E_{f,m,W}\Big]\\
&\leq \exp(d\varepsilon_0)\cdot\mathop{\mathbb{P}}_{\eta[j]\sim\mathsf{round}(\mathbf{N}(0,m/4))}\Big[\eta\in E_{f,m,W}\Big]+d\delta_0 && \text{(Basic composition \& Thm. 3.4)}\\
&=\exp(d\varepsilon_0)\cdot\mathbb{P}\Big[\widetilde{\mathsf{GM}}_{f,m/4}(W)\in E\Big]+d\delta_0 && \text{(Defn. of } \widetilde{\mathsf{GM}}_{f,m/4})\\
&=\exp(d\varepsilon_0)\cdot\mathbb{P}\Big[\mathsf{round}(\mathsf{GM}_{f,m/4}(W))\in E\Big]+d\delta_0 && (7)
\end{aligned}$$

The last equality comes from our version of De Moivre-Laplace (Lemma 3.3).

Recall that differential privacy is closed under post-processing. Because round is a data-independent function and Theorem 3.2 asserts that $\mathsf{GM}_{f,m/4}$ is (ε_1,δ_1)-differentially private,

$$\begin{aligned}
(7)&\leq \exp(d\varepsilon_0+\varepsilon_1)\cdot\mathbb{P}\Big[\mathsf{round}(\mathsf{GM}_{f,m/4}(W'))\in E\Big]+d\delta_0+\exp(d\varepsilon_0)\cdot\delta_1\\
&=\exp(d\varepsilon_0+\varepsilon_1)\cdot\mathbb{P}\Big[\widetilde{\mathsf{GM}}_{f,m/4}(W')\in E\Big]+d\delta_0+\exp(d\varepsilon_0)\cdot\delta_1 && \text{(Lemma 3.3)}\\
&=\exp(d\varepsilon_0+\varepsilon_1)\cdot\mathop{\mathbb{P}}_{\eta[j]\sim\mathsf{round}(\mathbf{N}(0,m/4))}\Big[\eta\in E_{f,h,W'}\Big]\\
&\quad+d\delta_0+\exp(d\varepsilon_0)\cdot\delta_1 && \text{(Defn. of } \widetilde{\mathsf{GM}}_{f,m/4})\\
&\leq \exp(2d\varepsilon_0+\varepsilon_1)\cdot\mathop{\mathbb{P}}_{\eta[j]\sim\mathbf{Bin}(h,1/2)}\Big[\eta\in E_{f,h,W'}\Big]\\
&\quad+\exp(d\varepsilon_0+\varepsilon_1)\cdot d\delta_0+d\delta_0+\exp(d\varepsilon_0)\cdot\delta_1 && \text{(Basic composition \& Thm. 3.4)}\\
&=\exp(2d\varepsilon_0+\varepsilon_1)\cdot\mathbb{P}\Big[\mathsf{BM}_{f,h}(W')\in E\Big]\\
&\quad+\exp(d\varepsilon_0+\varepsilon_1)\cdot d\delta_0+d\delta_0+\exp(d\varepsilon_0)\cdot\delta_1\\
&<\exp(2d\varepsilon_0+\varepsilon_1)\cdot\mathbb{P}\Big[\mathsf{BM}_{f,h}(W')\in E\Big]\\
&\quad+2\exp(d\varepsilon_0+\varepsilon_1)\cdot d\delta_0+\exp(d\varepsilon_0)\cdot\delta_1
\end{aligned}$$

This concludes the proof. □

Lemma 3.9. *For any* $X_i\in\mathcal{B}^d$, *if* $Y_i\leftarrow\mathsf{PRE}_{g,b,\tau}(X_i)$, *then* $\|Y_i\|_2\leq g/2+\sqrt{d}+\tau$ *with probability 1.*

Proof. Clients enforce $\left\|\eta_i\right\|_2\leq\tau$, so it will suffice to upper bound $\|W_i\|_2$ and then invoke the triangle

inequality.

$$\begin{aligned}
\|W_i\|_2^2 &= \sum_{j\in[d]} W_i[j]^2 \\
&\le \sum_{j\in[d]} (\hat{X}_i[j]+1)^2 \\
&= \sum_{j\in[d]} \hat{X}_i[j]^2 + 2\hat{X}_i[j] + 1 \\
&= \frac{g^2}{4}\left(\sum_{j\in[d]} X_i[j]^2\right) + g\left(\sum_{j\in[d]} X_i[j]\right) + d \\
&\le \frac{g^2}{4} + g\left(\sum_{j\in[d]} X_i[j]\right) + d && (X_i \in \mathcal{B}^d) \\
&\le \frac{g^2}{4} + g\|X_i\|_1 + d \\
&\le \frac{g^2}{4} + g\sqrt{d} + d
\end{aligned}$$

Thus, $\|W_i\|_2 \le g/2 + \sqrt{d}$ so that $\left\|Y_i = W_i + \eta_i\right\| \le g/2 + \sqrt{d} + \tau$. □

Lemma 3.14 (Parameters for DBM's Privacy)**.** *Fix $t \le n/6$, $\varepsilon \in (0, 9/10)$, and $\delta \in (0, 2e^{-6})$. Let $\varepsilon_{\text{priv}} \leftarrow 99\varepsilon/100$, $\varepsilon_{\text{sim}} \leftarrow \varepsilon/200d$, $\delta_{\text{priv}} \leftarrow \delta/5e^{\varepsilon}$, and $\delta_{\text{sim}} \leftarrow \delta/5de^{\varepsilon}$. When $b \ge \frac{12}{n\varepsilon_{\text{sim}}^2}\ln^2\frac{2}{\delta_{\text{sim}}}$, $g \le \varepsilon_{\text{priv}}\sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}} - 2\sqrt{d}$, and $\tau \ge \sqrt{\frac{db}{2}\ln(2nd/\delta_{\text{priv}})}$, $\Pi_{g,b,\tau,r}$ is $\left(\varepsilon\cdot\sqrt{\frac{n}{h}},\ \delta\cdot\exp\left(\varepsilon\cdot\sqrt{\frac{n}{h}} - \varepsilon\right)\right)$-differentially private. When all clients are honest ($h = n$), this simplifies to (ε, δ)-DP.*

Proof. For all attacks K, we need to show that the composite algorithm $\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}^K_{g,b,\tau}$ is differentially private with the desired parameters. Because addition is commutative and the honest clients' coins are independent of the malicious clients' coins, we can interpret $\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}^K_{g,b,\tau}(X)$ as a post-processing of $\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau}(X_{\mathcal{H}})$ which is the sum of the honest clients' messages that pass the inBall_r norm test. Due to the closure of differential privacy under post-processing, it will suffice to prove $\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau}$ is differentially private with the desired parameters.

Let Σ be the algorithm that takes a sequence of $\mathbb{F}^d$ values as input and reports their sum (aggregation without input certification). Because we have set r to be the maximum value $\left\|\mathsf{PRE}_{g,b,\tau}\right\|_2$ can take, observe that $(\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau})(X_{\mathcal{H}})$ is identically distributed with $(\Sigma \circ \mathsf{PRE}_{g,b,\tau})(X_{\mathcal{H}})$ for all inputs $X_{\mathcal{H}}$.

Now,

$$\begin{aligned}
\mathbb{P}\left[(\Sigma \circ \mathsf{PRE}_{g,b,\tau})(X_{\mathcal{H}}) \in E\right] &\le \mathbb{P}\left[(\Sigma \circ \mathsf{PRE}_{g,b,\infty})(X_{\mathcal{H}}) \in E\right] + \delta_{\text{priv}} && \text{(Lemma 3.13)} \\
&= \mathbb{P}\left[\mathsf{BM}_{\Sigma,h=hb}(W_{\mathcal{H}}) \in E\right] + \delta_{\text{priv}} && (8)
\end{aligned}$$

The second step comes from the fact that, for every $i \in \mathcal{H}$, the randomizer execution $\mathsf{PRE}_{g,b,\infty}(X_i)$ is the addition of $\mathbf{Bin}(b, 1/2)$ noise to the encoding W_i. Adding all of these up results in $\mathbf{Bin}(hb, 1/2)$ noise.

We will show that for $\varepsilon_0 \leftarrow \varepsilon_{\text{sim}}\cdot\sqrt{\frac{n}{h}}$, $\varepsilon_1 \leftarrow \varepsilon_{\text{priv}}\cdot\sqrt{\frac{n}{h}}$, $\delta_0 \leftarrow \delta_{\text{sim}}$, and $\delta_1 \leftarrow \delta_{\text{priv}}$, the following inequality holds:

$$hb \ge \max\left(\frac{12}{\varepsilon_0^2}\ln^2\frac{2}{\delta_0},\ \frac{8\Delta^2}{\varepsilon_1^2}\ln\frac{5}{4\delta_1}\right),$$

Because of our bounds on $t = n - h$ and ε, both $\varepsilon_0, \varepsilon_1$ lie within $(0,1)$. Theorem 3.5 can now be invoked:

$$
\begin{aligned}
(8) &\leq \exp(2d\varepsilon_0 + \varepsilon_1) \cdot \mathbb{P}\left[\mathsf{BM}_{\Sigma,h=hb}(W'_{\mathcal{H}}) \in E\right] \\
&\quad + 2\exp(d\varepsilon_0 + \varepsilon_1) \cdot d\delta_0 + \exp(d\varepsilon_0) \cdot \delta_1 + \delta_{\text{priv}} \\
&\leq \exp(2d\varepsilon_0 + \varepsilon_1) \cdot \mathbb{P}\left[(\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau})(X'_{\mathcal{H}}) \in E\right] \\
&\quad + 2\exp(d\varepsilon_0 + \varepsilon_1) \cdot d\delta_0 + \exp(d\varepsilon_0) \cdot \delta_1 + \exp(2d\varepsilon_0 + \varepsilon_1) \cdot \delta_{\text{priv}} + \delta_{\text{priv}} && \text{(Lemma 3.13)} \\
&< \exp\left(\sqrt{\frac{n}{h}} \cdot \varepsilon\right) \cdot \mathbb{P}\left[(\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau})(X'_{\mathcal{H}}) \in E\right] \\
&\quad + 2\exp(2d\varepsilon_0 + \varepsilon_1) \cdot d\delta_{\text{sim}} + 3\exp(d\varepsilon_0) \cdot \delta_{\text{priv}} && \text{(Choices of } \varepsilon_0, \varepsilon_1, \delta_0, \delta_1) \\
&< \exp\left(\sqrt{\frac{n}{h}} \cdot \varepsilon\right) \cdot \mathbb{P}\left[(\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau})(X'_{\mathcal{H}}) \in E\right] + \exp(2d\varepsilon_0 + \varepsilon_1 - \varepsilon) \cdot \delta && \text{(Choices of } \delta_{\text{sim}}, \delta_{\text{priv}}) \\
&= \exp\left(\sqrt{\frac{n}{h}} \cdot \varepsilon\right) \cdot \mathbb{P}\left[(\Sigma_{\mathsf{inBall}_r} \circ \mathsf{PRE}_{g,b,\tau})(X'_{\mathcal{H}}) \in E\right] + \exp(\varepsilon \cdot \sqrt{\frac{n}{h}} - \varepsilon) \cdot \delta && \text{(Choices of } \varepsilon_0, \varepsilon_1)
\end{aligned}
$$

which is the desired privacy guarantee.

Now, we prove $hb \geq \frac{12}{\varepsilon_0^2} \ln^2 \frac{2}{\delta_0}$.

$$
\begin{aligned}
& hb \\
&= \frac{h}{n} \cdot nb \\
&\geq \frac{h}{n} \cdot \frac{12}{\varepsilon_{\text{sim}}^2} \ln^2 \frac{2}{\delta_{\text{sim}}} && \text{(Range of } b) \\
&= \frac{12}{\varepsilon_0^2} \ln^2 \frac{2}{\delta_0}
\end{aligned}
$$

We finally prove $hb \geq \frac{8}{\varepsilon_1^2} \ln \frac{5}{4\delta_1}$:

$$
\begin{aligned}
& hb \\
&= \frac{h}{n} \cdot nb \\
&= \frac{h}{n} \cdot (\sqrt{nb})^2 \\
&= \frac{h}{n} \cdot \left(\varepsilon_{\text{priv}} \sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}}\right)^2 \cdot \frac{8}{\varepsilon_{\text{priv}}^2} \cdot \ln \frac{5}{4\delta_{\text{priv}}} \\
&= \frac{h}{n} \cdot \left(\varepsilon_{\text{priv}} \sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}} - 2\sqrt{d} + 2\sqrt{d}\right)^2 \cdot \frac{8}{\varepsilon_{\text{priv}}^2} \cdot \ln \frac{5}{4\delta_{\text{priv}}} \\
&\geq \frac{h}{n} \cdot \left(g + 2\sqrt{d}\right)^2 \cdot \frac{8}{\varepsilon_{\text{priv}}^2} \cdot \ln \frac{5}{4\delta_{\text{priv}}} && \text{(Range of } g) \\
&= \frac{h}{n} \cdot \frac{8}{\varepsilon_{\text{priv}}^2} \cdot \ln \frac{5}{4\delta_{\text{priv}}} && \text{(Lemma 3.10)} \\
&= \frac{8}{\varepsilon_1^2} \cdot \ln \frac{5}{4\delta_1}
\end{aligned}
$$

This concludes the proof. □

Lemma 3.12 (Mean Squared-Error of DBM). *Fix any attack K for t malicious clients. If $\mu^K \leftarrow \Pi^K_{g,b,\tau,r}(X)$, then the mean squared-error of μ^K with respect to $\mu \leftarrow \frac{1}{n}\sum_{i\in[n]} X_i$ is*

$$\mathbb{E}\left[\left\|\mu^K - \mu\right\|_2^2\right] \le \frac{4t^2}{n^2}\cdot(1+\sqrt{d}/g+\tau/g)^2 + \frac{d}{n^2}\cdot\frac{h(b+1)}{g^2}$$

When all clients are honest ($h = n$, $t = 0$), this simplifies to $\frac{d}{4ng^2}\cdot(b+1)$

Proof. When Π is under attack $K = (C, \mathbf{T})$, recall that $\{y_i^K\}_{i\in\mathcal{C}} \sim \mathbf{T}$ while honest clients independently sample $y_i^K \sim \mathsf{PRE}(X_i)$. Linearity of expectation implies

$$\mathop{\mathbb{E}}_{\mathsf{PRE}_{g,b,\tau},\mathbf{T}}\left[\left\|\mu^K - \mu\right\|_2^2\right] = \sum_{j\in[d]} \mathbb{E}\left[\mu^K[j]-\mu[j]\right]^2 + \mathrm{Var}\left[\mu^K[j]\right] \tag{9}$$

so we bound the bias and variance separately. Let y^K_{agg} denote the output of $(\Sigma_{\mathsf{inBall}_r}\circ\mathsf{PRE}^K_{g,b,\tau})(X)$. Because $\mathsf{POST}_g(y^K_{\mathrm{agg}}) = \frac{2}{ng}\cdot y^K_{\mathrm{agg}}$, the variance is

$$\begin{aligned}
&\mathrm{Var}\left[\mu^K[j]\right]\\
&= \frac{4}{n^2g^2}\cdot\mathrm{Var}\left[y^K_{\mathrm{agg}}[j]\right]\\
&= \frac{4}{n^2g^2}\cdot\mathrm{Var}\left[\sum_{i\in[n]}\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K[j]\right] && \text{(Defn. of } \Sigma_{\mathsf{inBall}_r}\text{)}\\
&= \frac{4}{n^2g^2}\cdot\left(\mathrm{Var}\left[\sum_{i\in\mathcal{C}}\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K[j]\right] + \mathrm{Var}\left[\sum_{i\in\mathcal{H}}\mathsf{inBall}_r(Y_i^K[j])\cdot Y_i^K[j]\right]\right) && (10)
\end{aligned}$$

The last step follows from the independence of the honest clients from the malicious clients.

By Lemma 3.9 and our choice of $r \leftarrow g/2+\sqrt{d}+\tau$, $\mathsf{inBall}_r(Y_i^K) = 1$ for any honest Y_i. Hence,

$$\begin{aligned}
(10) &= \frac{4}{n^2g^2}\cdot\left(\mathrm{Var}\left[\sum_{i\in\mathcal{C}}\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K[j]\right] + \sum_{i\in\mathcal{H}}\mathrm{Var}\left[W_i[j]+\eta_i[j]\right]\right)\\
&\le \frac{4}{n^2g^2}\cdot\left(\mathrm{Var}\left[\sum_{i\in\mathcal{C}}\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K[j]\right] + \frac{h}{4}(b+1)\right)
\end{aligned}$$

We unpack the last step. $W_i[j]$ is the sum of g bits but only one of them is random, so its variance is at most $1/4$. $\mathrm{Var}[\eta_i[j]]$ is bounded by the variance of the binomial noise before truncation, $b/4$.

Meanwhile, the bias is

$$\begin{aligned}
&\mathbb{E}\left[\mu^K[j]-\mu[j]\right]\\
&= \mathbb{E}\left[\sum_{i\in[n]}\frac{2}{ng}\cdot\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K[j] - \sum_{i\in[n]}\frac{1}{n}\cdot X_i[j]\right] && \text{(Defn. of } \mu,\mu^K\text{)}\\
&= \mathbb{E}\left[\sum_{i\in\mathcal{C}}\frac{2}{ng}\cdot\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K[j] - \frac{1}{n}\cdot X_i[j]\right]
\end{aligned}$$

The last equality comes from the lack of bias in any honestly generated message (Lemma 3.8) and the fact that every honestly generated message will have norm $\le r$.

By substituting our variance and bias bounds into (9), we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathsf{PRE}_{g,b,\tau},\mathbf{T}}\left[\left\|\mu^K-\mu\right\|_2^2\right]\\
&\le \sum_{j\in[d]}\mathbb{E}\left[\left(\sum_{i\in\mathcal{C}}\frac{2}{ng}\cdot\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K[j]-\frac{1}{n}\cdot X_i[j]\right)^2\right]+\frac{d}{n^2}\cdot\frac{h(b+1)}{g^2}\\
&=\mathbb{E}\left[\left\|\sum_{i\in\mathcal{C}}\frac{2}{ng}\cdot\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K-\frac{1}{n}\cdot X_i\right\|_2^2\right]+\frac{d}{n^2}\cdot\frac{h(b+1)}{g^2}\\
&\le\mathbb{E}\left[\left(\left\|\sum_{i\in\mathcal{C}}\frac{2}{ng}\cdot\mathsf{inBall}_r(Y_i^K)\cdot Y_i^K\right\|_2+\left\|\frac{1}{n}\sum_{i\in\mathcal{C}}X_i\right\|_2\right)^2\right] && \text{(Triangle ineq.)}\\
&\le\frac{t^2}{n^2}\cdot\left(\frac{2r}{g}+1\right)^2+\frac{d}{n^2}\cdot\frac{h(b+1)}{g^2}
\end{aligned}$$

In the above, we used the definition of inBall_r and the fact that $|\mathcal{C}|=t$. Our proof is complete by substituting our choice of r. □

Theorem 3.6. *For $\varepsilon\in(0,9/10)$ and $\delta\in(0,2e^{-6})$, there are choices of parameters g,b,τ,r such that $\Pi_{g,b,\tau,r}=(\mathsf{PRE}_{g,b,\tau},\mathsf{inBall}_r,\mathsf{POST}_g)$ ensures (ε,δ)-DP and solves DME with variance $O(\frac{d}{\varepsilon^2n^2}\log\frac{e^\varepsilon}{\delta})$ when there are no malicious clients. When under any attack by $t\le n/6$ clients, it ensures $\left(\varepsilon\cdot\sqrt{\frac{n}{n-t}},\ \delta\cdot\exp\left(\varepsilon\cdot\sqrt{\frac{n}{n-t}}-\varepsilon\right)\right)$-differential privacy and solves DME with mean-squared error*

$$\frac{t^2}{n^2}\cdot O\left(1+\frac{\sqrt{d}}{\varepsilon\sqrt{n}}\log\frac{e^\varepsilon nd}{\delta}\right)^2+\frac{n-t}{n}\cdot O\left(\frac{d}{\varepsilon^2n^2}\log\frac{e^\varepsilon}{\delta}\right)$$

Bias is only present due to attacks. It suffices for certified aggregation to operate on numbers in a field of size

$$n\cdot(g+b)=\tilde{O}\left(nd\log\frac{d}{\delta}+\frac{d^2}{\varepsilon^2}\log^2\frac{d}{\delta}\right)$$

Proof. We follow Lemma 3.14 and assign $g\leftarrow\lfloor\varepsilon_{\text{priv}}\sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}}-2\sqrt{d}\rfloor$, and $\tau\leftarrow\left\lceil\sqrt{\frac{db}{2}\ln(2nd/\delta_{\text{priv}})}\right\rceil$. Finally, we set $b\leftarrow 8+\lceil\frac{12}{n\varepsilon_{\text{sim}}^2}\ln^2\frac{2}{\delta_{\text{sim}}}\rceil$.

We make the observation that the error bound in Lemma 3.12 has multiple terms with g in the denominator. So to obtain an upper bound on the error, we will derive a lower bound on g.

$$\begin{aligned}
g&=\lfloor\varepsilon_{\text{priv}}\sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}}-2\sqrt{d}\rfloor\\
&\ge\varepsilon_{\text{priv}}\sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}}-2\sqrt{d}-1\\
&=\underbrace{\frac{99}{100}\varepsilon_{\text{priv}}\sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}}}_{T_1}+\underbrace{\frac{1}{100}\varepsilon_{\text{priv}}\sqrt{\frac{nb}{8\ln(5/4\delta_{\text{priv}})}}-2\sqrt{d}-1}_{T_2}
\end{aligned}$$

We now show that $T_2 > 0$, so that $g > T_1$:

$$\begin{aligned}
T_2 &\geq \frac{\varepsilon_{\text{priv}}}{100\varepsilon_{\text{sim}}}\sqrt{\frac{3\ln^2(2/\delta_{\text{sim}})}{2\ln(5/4\delta_{\text{priv}})}} - 2\sqrt{d} - 1 \\
&\geq \varepsilon_{\text{priv}} \cdot \frac{2d}{\varepsilon} - 2\sqrt{d} - 1 \\
&= \frac{99}{100} \cdot 2d - 2\sqrt{d} - 1 \\
&= \frac{99}{50}d - 2\sqrt{d} - 1 \\
&> 0 && (d > 1)
\end{aligned}$$

We finally bound τ/g, $\sqrt{d}/g$, and $(b+1)/g^2$ to substitute into Lemma 3.12.

$$\begin{aligned}
\frac{\tau}{g} &\leq \frac{1}{g} \cdot \left(\sqrt{\frac{db}{2}\ln\frac{2nd}{\delta_{\text{priv}}}} + 1\right) \\
&\leq \frac{100}{99\varepsilon_{\text{priv}}} \cdot \sqrt{\frac{8}{nb}\ln\frac{5}{4\delta_{\text{priv}}}} \cdot \left(\sqrt{\frac{db}{2}\ln\frac{2nd}{\delta_{\text{priv}}}} + 1\right) \\
&< \frac{100}{99\varepsilon_{\text{priv}}} \cdot \left(\frac{2\sqrt{d}}{\sqrt{n}}\ln\frac{2nd}{\delta_{\text{priv}}} + \sqrt{\frac{8}{nb}\ln\frac{5}{4\delta_{\text{priv}}}}\right) \\
&< \frac{300}{99\varepsilon_{\text{priv}}} \cdot \sqrt{\frac{d}{n}} \cdot \ln\frac{2nd}{\delta_{\text{priv}}} && (b > 8) \\
&= \frac{3 \cdot 10^4}{99^2\varepsilon} \cdot \sqrt{\frac{d}{n}} \cdot \ln\frac{10e^{\varepsilon}nd}{\delta}
\end{aligned}$$

The last inequality comes from our choice of $\varepsilon_{\text{priv}}, \delta_{\text{priv}}$. Meanwhile,

$$\begin{aligned}
\frac{\sqrt{d}}{g} &\leq \frac{100}{99\varepsilon_{\text{priv}}} \cdot \sqrt{\frac{8d\ln(5/4\delta_{\text{priv}})}{nb}} \\
&\leq \frac{100\varepsilon_{\text{sim}}}{99\varepsilon_{\text{priv}}} \cdot \sqrt{\frac{2d\ln(5/4\delta_{\text{priv}})}{3\ln^2(2/\delta_{\text{sim}})}} \\
&= \frac{\varepsilon}{99 \cdot 2\varepsilon_{\text{priv}}} \cdot \sqrt{\frac{2\ln(5/4\delta_{\text{priv}})}{3d\ln^2(2/\delta_{\text{sim}})}} \\
&= \frac{50}{99^2} \cdot \sqrt{\frac{2\ln(5/4\delta_{\text{priv}})}{3d\ln^2(2/\delta_{\text{sim}})}} \\
&< \frac{50}{99^2} \cdot \sqrt{\frac{2}{3d}}
\end{aligned}$$

Finally,

$$\begin{aligned}
\frac{b+1}{g^2} &\leq \frac{100^2}{99^2\varepsilon_{\text{priv}}^2} \cdot \frac{8\ln(5/4\delta_{\text{priv}})}{nb} \cdot (b+1) \\
&= \frac{100^2}{99^2\varepsilon_{\text{priv}}^2} \cdot \frac{8\ln(5/4\delta_{\text{priv}})}{n} + \frac{100^2}{99^2\varepsilon_{\text{priv}}^2} \cdot \frac{8\ln(5/4\delta_{\text{priv}})}{nb} \\
&< \frac{100^2}{99^2\varepsilon_{\text{priv}}^2} \cdot \frac{9\ln(5/4\delta_{\text{priv}})}{n} && (b > 8) \\
&= \frac{100^4}{99^4\varepsilon^2} \cdot \frac{9\ln(25e^{\varepsilon}/4\delta)}{n}
\end{aligned}$$

□

we visualize the error bound in Figure 6 below, in a regime where $d < n$ and a regime where $d > n$.

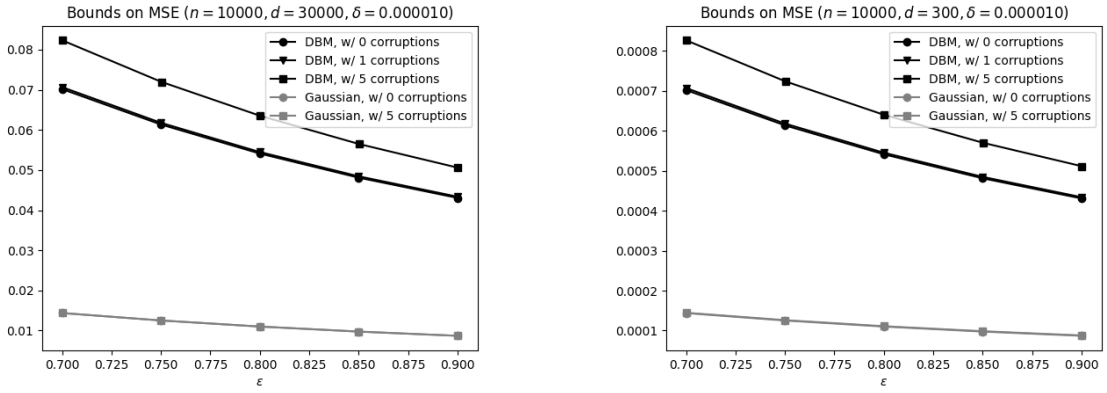

Figure 6: Visualization of our bound on mean-squared error of our distributed binomial mechanism (DBM) as a function of privacy parameter ε (smaller is better). We plot the corresponding bound for the Gaussian mechanism for comparison.

E Composition of DBM

E.0.1 Repeated Executions of the DBM

We have presented a bound on the expected squared error of one usage of the DBM for private mean estimation. But an application like gradient descent will require many executions of the DBM. Although its guarantee of differential privacy will remain, the parameters will grow over time. Re-scaling parameters to account for this unavoidable phenomenon will inflate the expected squared error of any given mean estimate. The field size will also increase. Bounding the rate(s) of growth is therefore of practical importance.

Fortunately, our protocol is structured in a way that is amenable to privacy accounting over R rounds. At a high-level, we use a two step process: we apply composition theorems for the Gaussian mechanism over R rounds and then argue R executions of our protocol is a good simulation thereof. Approximate DP composition is only used to account for the fidelity of the Gaussian noise simulation: increasing the standard deviation σ compensates for loss in fidelity accumulated over the R repetitions but does not impact estimation error, as we always match the granularity of discretization to σ and the output party re-scales by the granularity.

In this Appendix, we provide tools to analyze the privacy guarantee of the R-fold composition of the DBM. For clarity, we repeat the intuition given in the main text. We use a two step process: we apply composition theorems for the Gaussian mechanism over R rounds and then argue R executions of our protocol is a good simulation thereof. Approximate DP composition is only used to account for the fidelity of the Gaussian noise simulation: increasing the standard deviation σ compensates for loss in fidelity accumulated over the R repetitions but does not impact estimation error, as we always match the granularity of discretization to σ and the output party re-scales by the granularity.

More precisely, we first derive a variant of Theorem 3.2: R repetitions of the Gaussian mechanism GM_{f,σ^2} will satisfy (ε,δ)-DP provided that σ^2 is linear in R. This result is obtained by utilizing the composition theorem for Gaussian Differential Privacy—see Dong Roth and Su [28]—and then translating into approximate differential privacy. Next, we invoke our version of De Moivre-Laplace (Theorem 3.4) to obtain an approximate DP guarantee for R-fold composition of the binomial mechanism $\mathsf{BM}_{f,h}$. The result is a variant of Theorem 3.5 where d increases to $d \cdot R$ and $\frac{8\Delta^2}{\varepsilon_1^2} \ln \frac{5}{4\delta_1}$ increases to $\frac{8R\Delta^2}{\varepsilon_1^2} \ln \frac{5}{4\delta_1}$. The rest of our analysis can be replicated with these re-scaled values.

We leverage the work by Dong et al. on functional[17] differential privacy and Gaussian differential privacy. Except for the final two theorems, all statements in this appendix come from Dong et al.

Definition E.1 (Tradeoff Functions)**.** For any distributions $\mathbf{P}, \mathbf{Q}$, the tradeoff function between them is

$$T_{\mathbf{P},\mathbf{Q}}(\alpha) := \inf_{\theta}\{\beta_\theta \mid \alpha_\theta \leq \alpha\}$$

where $\alpha_\theta, \beta_\theta$ are the type I and type II error rates of rejection rule θ when hypothesis testing between $\mathbf{P}, \mathbf{Q}$.

Definition E.2 (Functional and Gaussian DP)**.** Let $g : [0,1] \to [0,1]$ be a convex, continuous, and non-increasing function. Algorithm M satisfies g-DP if, for all neighboring X, X', $T_{M(X),M(X')}(\alpha) \geq g(\alpha)$. In the case where $g = T_{\mathbf{N}(0,1),\mathbf{N}(\mu,1)}$, we say M satisfies μ-Gaussian DP.

Theorem E.3 (Theorem 2.7 in [28])**.** *For Δ-sensitive function f, the Gaussian mechanism GM_{f,σ^2} satisfies Δ/σ-GDP.*

Lemma E.4 (Corollary 3.3 in [28])**.** *The R-fold composition of a μ-GDP algorithm satisfies $\sqrt{R} \cdot \mu$-GDP.*

Let $\mathsf{GM}^R_{f,\sigma^2}$ be the R-fold composition of the Gaussian mechanism GM_{f,σ^2}. From the preceding two statements, we know that $\mathsf{GM}^R_{f,\sigma^2}$ satisfies $\sqrt{R} \cdot \Delta/\sigma$-GDP. But in order to obtain a privacy guarantee for the R-fold composition of the binomial mechanism, we require an approximate DP guarantee.

Theorem E.5. *For Δ-sensitive function f and $\sigma^2 \geq R \cdot \frac{2\Delta^2}{\varepsilon^2} \ln \frac{5}{4\delta}$, $\mathsf{GM}^R_{f,\sigma^2}$ satisfies (ε,δ)-DP.*

Proof. As established, $\mathsf{GM}^R_{f,\sigma^2}$ satisfies $\sqrt{R} \cdot \Delta/\sigma$-GDP. By definition, this means its tradeoff curve lies above $T(\mathbf{N}(0,1), \mathbf{N}(\sqrt{R} \cdot \Delta/\sigma, 1))$. Observe that this second tradeoff curve is that of the Gaussian mechanism for a function $\hat{f}$ with sensitivity $\sqrt{R} \cdot \Delta/\sigma$ and with variance $\hat{\sigma}^2 = 1$. Due to our lower bound on σ^2 and Theorem 3.2, we know that $\mathsf{GM}_{\hat{f},\hat{\sigma}^2}$ satisfies (ε,δ)-dp.

We rely on the following technical lemma:

Lemma E.6 (From [66])**.** *Define $g_{\varepsilon,\delta}(\alpha) := \max(0, 1-\delta-e^\varepsilon\alpha, e^{-\varepsilon}(1-\delta-\alpha))$. An algorithm satisfies (ε,δ)-differential privacy if and only if it satisfies $g_{\varepsilon,\delta}$-DP.*

Hence $T(\mathbf{N}(0,1), \mathbf{N}(\sqrt{R} \cdot \Delta/\sigma, 1))$ lies above $g_{\varepsilon,\delta}$. Transitivity implies that the tradeoff curve for $\mathsf{GM}^R_{f,\sigma^2}$ lies above $g_{\varepsilon,\delta}$, which in turn means our protocol is (ε,δ)-DP. □

We combine the above result with our version of De Moivre-Laplace (Theorem 3.4) to obtain a privacy guarantee for R-fold composition of $\mathsf{BM}_{f,h}$.

[17] Called "f"-differential privacy by the authors.

Theorem E.7 (Privacy of $\mathsf{BM}^R_{f,h}$)**.** *Fix any ε_0, ε_1, δ_0, and δ_1 in the interval $(0,1)$. If h is an even integer such that $h > \max\left(\frac{12}{\varepsilon_0^2}\ln^2\frac{2}{\delta_0}, \frac{8R\Delta^2}{\varepsilon_1^2}\ln\frac{5}{4\delta_1}\right)$, then for any event E,*

$$\mathbb{P}\left[\mathsf{BM}^R_{f,h}(W)\in E\right] \le \exp(2dR\varepsilon_0+\varepsilon_1)\cdot\mathbb{P}\left[\mathsf{BM}_{f,h}(W')\in E\right] + 2\exp(dR\varepsilon_0+\varepsilon_1)\cdot d\delta_0 + \exp(d\varepsilon_0)\cdot\delta_1$$

The derivation is almost identical to that of Theorem 3.5, except that we need to invoke Theorem 3.4 dR times instead of d times—as there are more Gaussians that need to be simulated—and invoke Theorem E.5 instead of Theorem 3.2.

F Proofs for Lower Bound

Lemma 5.3. *Fix arbitrary $\hat{d}\in\mathbb{N}$. Suppose $\Pi = (\mathsf{PRE}, P, \mathsf{POST})$ is a protocol that wraps around certified secure aggregation Π_{Agg} and $\mathsf{POST}(y) := yQ$ for some public matrix $Q\in\mathbb{R}^{\hat{d}\times d}$. If $\Pi(\mathbf{D}^n)$ is an estimate of $\mathbb{E}[\mathbf{D}]$ such that the standard deviation of error in any coordinate j exceeds the bias in j, then there exists a vector $\tilde{v}\in\mathbb{F}^{\hat{d}}$ such that*

$$\left\|f(\tilde{v})\cdot\tilde{v}Q - \mathop{\mathbb{E}}_{v\sim\mathsf{PRE}(\mathbf{D})}[P(v)\cdot vQ]\right\|_2^2 \ge \frac{\kappa}{2}\cdot\min\left(\frac{d}{\varepsilon^2n^3}, 1/n\right)$$

where κ is the constant from Theorem 5.2.

Proof. Our argument will be probabilistic: we will prove that, when u is drawn from $\mathsf{PRE}(\mathbf{D})$, the expected value of $\left\|P(u)\cdot uQ - \mathop{\mathbb{E}}_{v\sim\mathsf{PRE}(\mathbf{D})}[P(v)\cdot vQ]\right\|_2^2$ exceeds the threshold so there must exist some $\tilde{v}$ where the desired inequality holds. In the following, random variables are generated in an execution of Π (absent an attack) on input $X\sim\mathbf{D}^n$. $y_{\text{agg}} = \sum y_i\cdot P(y_i)$ is the certified aggregation received by the analyst.

$$\begin{aligned}
&\kappa\cdot\min\left(\frac{d}{\varepsilon^2n^2}, 1\right)\\
&\le \mathop{\mathbb{E}}_{y_{\text{agg}}}\left[\|\Pi(\mathbf{D}^n)-\mathbb{E}[\mathbf{D}]\|_2^2\right] && \text{(Theorem 5.2)}\\
&= \mathop{\mathbb{E}}_{y_{\text{agg}}}\left[\left\|y_{\text{agg}}Q-\mathbb{E}[\mathbf{D}]\right\|_2^2\right] && \text{(Assumption on } A\text{)}\\
&= \sum_{j\in[d]}\mathop{\mathbb{E}}_{y_{\text{agg}}}\left[\left((y_{\text{agg}}Q)[j]-\mathbb{E}[\mathbf{D}][j]\right)^2\right] && \text{(Linearity of expectation)}\\
&\le \sum_{j\in[d]} 2\mathrm{Var}\left[(y_{\text{agg}}Q)[j]\right] && \text{(Assumption on } \Pi\text{'s estimate)}\\
&= 2n\cdot\sum_{j\in[d]}\mathop{\mathrm{Var}}_{u\sim\mathsf{PRE}(\mathbf{D})}\left[\sum_{\hat{\jmath}\in[\hat{d}]}u[\hat{\jmath}]\cdot P(u)\cdot Q_{\hat{\jmath}}[j]\right] && \text{(Each } y_i\sim_{\text{iid}}\mathsf{PRE}(\mathbf{D})\text{)}
\end{aligned}$$

By dividing by $2n$ and expanding the definition of variance,

$$\begin{aligned}
&\frac{\kappa}{2}\cdot\min\left(\frac{d}{\varepsilon^2n^3}, 1/n\right)\\
&\le \sum_{j\in[d]}\mathop{\mathbb{E}}_{u\sim\mathsf{PRE}(\mathbf{D})}\left[\left(\sum_{\hat{\jmath}\in[\hat{d}]}u[\hat{\jmath}]\cdot P(u)\cdot Q_{\hat{\jmath}}[j] - \mathop{\mathbb{E}}_{v\sim\mathsf{PRE}(\mathbf{D})}\left[\sum_{\hat{\jmath}\in[\hat{d}]}v[\hat{\jmath}]\cdot P(v)\cdot Q_{\hat{\jmath}}[j]\right]\right)^2\right]\\
&= \mathop{\mathbb{E}}_{u\sim\mathsf{PRE}(\mathbf{D})}\left[\left\|P(u)\cdot uQ - \mathop{\mathbb{E}}_{v\sim\mathsf{PRE}(\mathbf{D})}[P(v)\cdot vQ]\right\|_2^2\right]
\end{aligned}$$

By definition of expected value, there must be some $\tilde{v}$ which satisfies the desired inequality. This completes the proof. □

Theorem 5.1. *Fix arbitrary $\hat{d} \in \mathbb{N}$. Suppose $\Pi = (\mathsf{PRE}, P, \mathsf{POST})$ is an (ε, δ)-DP protocol that wraps around certified secure aggregation Π_{Agg} and $\mathsf{POST}(y) := yQ$ for some public matrix $Q \in \mathbb{R}^{\hat{d} \times d}$. If $\Pi(\mathbf{D}^n)$ is an estimate of $\mathbb{E}[\mathbf{D}]$ whose standard deviation of error in any coordinate j exceeds the bias in j, then there is an attack K such that*

$$\mathbb{E}\left[\left\|\Pi^K(\mathbf{D}^n) - \mathbb{E}[\mathbf{D}]\right\|_2^2\right] \geq c \cdot \min\left(\frac{t^2}{n^2} \cdot \frac{d}{\varepsilon^2 n}, \frac{t^2}{n}\right).$$

where c is a universal constant.

Proof. Suppose t malicious clients each send the $\tilde{v}$ message identified in Lemma 5.3. Formally, $\mathbf{P}$ is the distribution that places all mass on $\tilde{v}, \ldots, \tilde{v}$ and corrupt clients $\mathcal{C}$ is an arbitrary subset of $[n]$ with size t.

We will use y_{agg}^K) to denote the object received by the analyst in the attacked execution. The mean squared error of Π's output is

$$\begin{aligned}
&\mathbb{E}\left[\left\|\Pi^K(\mathbf{D}^n) - \mathbb{E}[\mathbf{D}]\right\|_2^2\right] \\
&= \sum_{j \in [d]} \mathbb{E}\left[\langle y_{\text{agg}}^K, Q[j]\rangle - \mathbb{E}[\mathbf{D}][j]\right]^2 && \text{(Assumption on } A\text{)} \\
&= \sum_{j \in [d]} \mathop{\mathbb{E}}_{y_i \sim_{\text{iid}} \mathsf{PRE}(\mathbf{D})} \left[\left\langle \sum_{i \in H} P(y_i) \cdot y_i, Q[j] \right\rangle + \left\langle \sum_{i \in \mathcal{C}} P(\tilde{v}) \cdot \tilde{v}, Q[j] \right\rangle - \mathbb{E}[\mathbf{D}][j]\right]^2 && \text{(Construction of } K = (\mathcal{C}, \mathbf{P})\text{)} \\
&= \sum_{j \in [d]} \mathop{\mathbb{E}}_{y_i \sim_{\text{iid}} \mathsf{PRE}(\mathbf{D})} \left[\left\langle \sum_{i \in [n]} P(y_i) \cdot y_i - \sum_{i \in \mathcal{C}} P(y_i) \cdot y_i, Q[j] \right\rangle + \left\langle \sum_{i \in \mathcal{C}} P(\tilde{v}) \cdot \tilde{v}, Q[j] \right\rangle - \mathbb{E}[\mathbf{D}][j]\right]^2 \\
&= \sum_{j \in [d]} \mathbb{E}\left[\left\langle \sum_{i \in \mathcal{C}} P(\tilde{v}) \cdot \tilde{v} - P(y_i) \cdot y_i, Q[j] \right\rangle\right]^2 && \text{(Π is unbiased)} \\
&= \sum_{j \in [d]} \mathbb{E}\left[\sum_{\hat{\jmath} \in [\hat{d}]} \sum_{i \in \mathcal{C}} (P(\tilde{v}) \cdot \tilde{v}[\hat{\jmath}] - P(y_i) \cdot y_i[\hat{\jmath}]) Q_{\hat{\jmath}}[j]\right]^2 \\
&= t^2 \sum_{j \in [d]} \mathop{\mathbb{E}}_{v \sim \mathsf{PRE}(\mathbf{D})} \left[\sum_{\hat{\jmath} \in [\hat{d}]} (P(\tilde{v}) \cdot \tilde{v}[\hat{\jmath}] - P(v) \cdot v[\hat{\jmath}]) Q_{\hat{\jmath}}[j]\right]^2 && (y_i \text{ are i.i.d.}) \\
&= t^2 \left\| P(\tilde{v}) \cdot \tilde{v} Q - \mathop{\mathbb{E}}_{v \sim \mathsf{PRE}(\mathbf{D})} [P(v) \cdot vQ] \right\|_2^2 \\
&\geq \frac{\kappa}{2} \cdot \min\left(\frac{t^2}{n^2} \cdot \frac{d}{\varepsilon^2 n}, \frac{t^2}{n}\right)
\end{aligned}$$

The last step comes from our preceding lemma. □

G Analysis of the Poisson-Binomial Mechanism (PBM)

In this Appendix, we analyze the privacy and accuracy guarantees of the PBM. The protocol was originally presented by Chen et al. [19] but we study its guarantees the presence of malicious clients. Much like the DBM, the PBM wraps around a generic certified aggregation protocol so it is specified by a client pre-processing algorithm, a predicate for the certified aggregation, and a post-processing algorithm run by the analyst.

As discussed in the introduction, the PBM ensures Rényi differential privacy assuming an ideal functionality that performs secure aggregation. The privacy definition is based upon the order-α Rényi divergence between distributions $\mathbf{P}, \mathbf{Q}$:

$$D_\alpha(\mathbf{P}, \mathbf{Q}) := \frac{1}{\alpha - 1} \log \mathop{\mathbb{E}}_{x \sim \mathbf{Q}} \left[\left(\frac{\mathbf{P}(x)}{\mathbf{Q}(x)} \right)^\alpha \right]$$

M is (α, ε)-Rényi differentially private if $D_\alpha(M(X), M(X')) \leq \varepsilon$ and $D_\alpha(M(X'), M(X)) \leq \varepsilon$ for any neighboring X, X'.

Before we describe the PBM for data belonging to the unit ball, we establish guarantees for data belonging to the interval $[-c, +c]$. These guarantees will extend to the higher-dimensional case in one of two different ways.

G.1 Poisson-Binomial Mechanism for Scalar Data

At a high level, the pre-processor samples a value from a binomial distribution whose maximum value is ℓ and whose expected value is a linear function of X_i. The certified aggregation $\Sigma_{\in[0,\ell]}$ will only add values in the range $[0, \ell]$. The post-processor can easily recover the mean of data from the mean of the pre-processed values by performing an inverse linear transformation.

The formal specification of the pre- and post-processing algorithms are below:

$$\mathsf{PRE}_{c,\theta,\ell}(X_i) = \mathbf{Bin}(\ell, p) \text{ for } p := \frac{1}{2} + \theta \cdot \frac{X_i}{c} \tag{11}$$

$$\mathsf{POST}_{c,\theta,\ell}(y_{\text{agg}}) = \frac{c}{\ell n \theta} \left(y_{\text{agg}} - \frac{\ell n}{2} \right) \tag{12}$$

Our main result concerning the PBM for scalar data is the following theorem:

Theorem G.1. *For any ε, α, there exist choices for $\ell \in \mathbb{N}$ and $\theta \in (0, 1/2)$ such that $\Pi_{c,\theta,\ell} = (\mathsf{PRE}_{c,\theta,\ell}, \in [0, \ell], \mathsf{POST}_{c,\theta,\ell})$ ensures $(\alpha,\ \varepsilon \cdot \frac{n}{n-t})$-RDP and solves DME with mean-squared error*

$$O\left(\frac{c^2 t^2}{n^2} \cdot \frac{\ell \alpha}{\varepsilon n} + \frac{n - t}{n} \cdot \frac{c^2 \alpha}{\varepsilon n^2} \right)$$

under any attack by t clients. It suffices for certified aggregation to operate on nmbers in a field of size $n \cdot \ell$.

G.1.1 Privacy Analysis

As with the DBM, we will focus on privacy offered against the analyst. When there is no attack, its view is formed by adding up n values generated by $\mathsf{PRE}_{c,\theta,\ell}$. The prior work shows that this has privacy guarantees improving with n. Formally,

Theorem G.2 (Implicit in Chen et al. [19])**.** *There is a constant κ such that, for $\varepsilon(\alpha, n) := \kappa \cdot \frac{\ell \theta^2}{n} \cdot \frac{1}{(1-2\theta)^4} \alpha$ the composition of $\Sigma_{[0,\ell]}$ and n executions of $\mathsf{PRE}_{c,\theta,\ell}$ satisfies $(\varepsilon(\alpha, n), \alpha)$-RDP.*

Although Chen et al. do not bound κ, they give an algorithm to compute $\varepsilon(\alpha, n)$. When $\alpha = 2$, we show that we can avoid divisions between small numbers. Refer to Appendix H for more detail.

We argue that the theorem extends to the malicious case. That is, the view of the analyst is simply the outcome of an RDP algorithm whose parameters depend on the number of malicious clients t.

Claim G.3. *If the number of malicious clients is t, then for any attack K the composed algorithm $\Sigma_{[0,\ell]} \circ R^K_{c,\theta,\ell}$ satisfies $(\varepsilon(\alpha, n - t), \alpha)$-RDP.*

Proof. Because addition is commutative and honest clients' coins are independent of malicious clients, we can interpret the contribution of malicious clients as a post-processing of the sum of honest clients' messages. The claim follows from Theorem G.2 and closure of differential privacy under post-processing. □

G.1.2 Accuracy Analysis

We now bound the mean-squared error in terms of the protocol's parameters, so that we can later substitute the parameters that target a certain level of privacy.

Claim G.4. *For any input $X \in [-c, c]^n$, let $\mu \leftarrow \frac{1}{n}\sum X_i$. For any attack K, if $\tilde{\mu} \leftarrow \Pi^K_{c,\theta,\ell}(X)$ then*

$$\mathbb{E}\left[(\tilde{\mu} - \mu)^2\right] \leq \frac{c^2 t^2}{n^2\theta^2}\left(\frac{1}{2} + \theta\right)^2 + \frac{n-t}{n} \cdot \frac{c^2}{4\ell n\theta^2}$$

In the special case where $t = 0$, the estimate is unbiased and the variance is $\leq \frac{c^2}{4\ell n\theta^2}$

Proof. We use $\hat{\mu}$ to denote the estimate generated by an honest execution of the protocol (no attack). Chen et al. proved that $\mathbb{E}[\hat{\mu}] = \mu$, so

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{P},\mathsf{PRE}_{c,\theta,\ell}}[\tilde{\mu} - \mu] &= \mathop{\mathbb{E}}_{\mathbf{P},\mathsf{PRE}_{c,\theta,\ell}}[\tilde{\mu} - \hat{\mu}] \\
&= \frac{c}{\ell n\theta} \cdot \left(\mathop{\mathbb{E}}_{\mathbf{P},\mathsf{PRE}_{c,\theta,\ell}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i + \sum_{i\in\mathcal{H}} y_i - \frac{\ell n}{2}\right] - \mathop{\mathbb{E}}_{\mathsf{PRE}_{c,\theta,\ell}}\left[\sum_{i=1}^{n} y_i - \frac{\ell n}{2}\right]\right) && \text{(Defn. of } \mathsf{POST}_{c,\theta,\ell}\text{)} \\
&= \frac{c}{\ell n\theta} \cdot \left(\mathop{\mathbb{E}}_{\mathbf{P}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i\right] - \mathbb{E}\left[\sum_{i\notin\mathcal{H}} y_i\right]\right) \\
&= \frac{c}{\ell n\theta} \cdot \left(\mathop{\mathbb{E}}_{\mathbf{P}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i\right] - \sum_{i\notin\mathcal{H}} \ell \cdot \left(\frac{1}{2} + \theta \cdot \frac{X_i}{c}\right)\right) && \text{(Defn. of } \mathsf{PRE}_{c,\theta,\ell}\text{)} \\
&= \frac{c}{\ell n\theta} \cdot \mathop{\mathbb{E}}_{\mathbf{P}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i - \ell \cdot \left(\frac{1}{2} + \theta \cdot \frac{X_i}{c}\right)\right]
\end{aligned}$$

Now we derive the variance:

$$\begin{aligned}
\mathrm{Var}[\tilde{\mu}] &= \frac{c^2}{\ell^2 n^2\theta^2}\left(\mathop{\mathrm{Var}}_{\mathbf{P}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i\right] + \sum_{i\in\mathcal{H}} \mathop{\mathrm{Var}}_{\mathsf{PRE}_{c,\theta,\ell}}[y_i]\right) && \text{(Independence)} \\
&= \frac{c^2}{\ell^2 n^2\theta^2}\left(\mathop{\mathrm{Var}}_{\mathbf{P}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i\right] + \ell \cdot \sum_{i\in\mathcal{H}} \frac{1}{4} - \theta^2 \cdot \frac{X_i^2}{c^2}\right) && \text{(Defn. of } \mathsf{PRE}_{c,\theta,\ell}\text{)} \\
&= \frac{c^2}{\ell^2 n^2\theta^2}\mathop{\mathrm{Var}}_{\mathbf{P}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i\right] + \frac{c^2}{\ell n^2\theta^2} \cdot \sum_{i\in\mathcal{H}} \frac{1}{4} - \theta^2 \cdot \frac{X_i^2}{c^2}
\end{aligned}$$

Thus, the mean squared error is, for any input X,

$$\begin{aligned}
&\mathbb{E}\left[(\tilde{\mu} - \mu)^2\right] \\
&= \mathrm{Var}[\tilde{\mu}] + \mathbb{E}[\tilde{\mu} - \mu]^2 \\
&= \frac{c^2}{\ell^2 n^2\theta^2} \cdot \left(\mathop{\mathrm{Var}}_{\mathbf{P}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i\right] + \mathop{\mathbb{E}}_{\mathbf{P}}\left[\sum_{i\notin\mathcal{H}} \tilde{y}_i - \ell \cdot \left(\frac{1}{2} + \theta \cdot \frac{X_i}{c}\right)\right]^2\right) + \frac{c^2}{\ell n^2\theta^2} \cdot \sum_{i\in\mathcal{H}} \frac{1}{4} - \theta^2 \cdot \frac{X_i^2}{c^2} \\
&= \frac{c^2}{\ell^2 n^2\theta^2}\mathop{\mathbb{E}}_{\mathbf{P}}\left[\left(\sum_{i\notin\mathcal{H}} \tilde{y}_i - \ell \cdot \left(\frac{1}{2} + \theta \cdot \frac{X_i}{c}\right)\right)^2\right] + \frac{c^2}{\ell n^2\theta^2} \cdot \sum_{i\in\mathcal{H}} \frac{1}{4} - \theta^2 \cdot \frac{X_i^2}{c^2} \qquad (13)
\end{aligned}$$

Now, we find an input X and choices for $\tilde{y}_i$ to maximize the above quantity. We maximize each term in the left-hand series by setting $\tilde{y}_i \leftarrow \ell$ and $X_{i\in[m]} \leftarrow -c$. We maximize each term in the right-hand series by setting $X_i \leftarrow 0$. So,

$$
\begin{aligned}
(13) &\leq \frac{c^2}{\ell^2 n^2 \theta^2} \mathbb{E}_{\mathbf{P}}\left[\left(\sum_{i \notin \mathcal{H}} \frac{\ell}{2} - \ell\theta \cdot \frac{X_i}{c}\right)\right]^2 + \frac{c^2}{\ell n^2 \theta^2} \cdot \sum_{i \in \mathcal{H}} \frac{1}{4} - \theta^2 \cdot \frac{X_i^2}{c^2} \\
&\leq \frac{c^2}{\ell^2 n^2 \theta^2}\left(\frac{m\ell}{2} + m\ell\theta\right)^2 + \frac{c^2}{\ell n^2 \theta^2} \cdot \frac{n-t}{4} \\
&= \frac{c^2 t^2}{n^2 \theta^2}\left(\frac{1}{2} + \theta\right)^2 + \frac{n-t}{n} \cdot \frac{c^2}{4\ell n \theta^2}
\end{aligned}
$$

□

G.2 Poisson-Binomial Mechanism for Data in Euclidean Unit Ball

The PBM for scalar data can be used to estimate the mean of vectors inside $\mathcal{B}^d$ by simply executing the mechanism on every coordinate (setting $c = 1$). But the privacy parameter degrades:

Theorem G.5 (Mironov [49])**.** *If M is (α, ε)-RDP, then the composition $M(X[1]), \ldots, M(X[d])$ is $(\alpha, d \cdot \varepsilon)$-RDP.*

To achieve a target level of privacy in an honest protocol execution, the term $\ell\theta^2$ needs to be shrunk by $\approx d$ (see Theorem G.2). This impacts our bound on the squared error. To compensate, we present two methods of reducing global sensitivity by $\approx \sqrt{d}$, as measured in ℓ_∞ norm. Both variants assume there is a matrix U accessible to all parties. We will use $U[t]$ to denote the t-th column vector. Meanwhile, U_j denotes the j-th row vector.

G.2.1 The Kashin Representation method

In this subsection, we review the way Chen et al. extend the PBM from the scalar case to the $\mathcal{B}^d$ case. The authors propose that each client maps their data X_i to a new point W_i whose number of dimensions is $D > d$ and whose ℓ_∞ norm is $c = \Theta(1/\sqrt{d})$. Then the clients simply execute the scalar Poisson-Binomial mechanism on each of the coordinates in W_i. The mapping is done with the *Kashin representation* algorithm by Lyubarskii and Vershynin [47].

Definition G.6 (Kashin Representations [47])**.** A matrix $U \in \mathbb{R}^{d \times D}$ with orthonormal rows *admits a S-Kashin representation of* $x \in \mathbb{R}^d$ if there is some column vector $w \in \mathbb{R}^D$ where $x = Uw$ and $\|w\|_\infty \leq \frac{S}{\sqrt{D}} \cdot \|x\|_2$. Matrix U is *S-Kashin* if it admits a S-Kashin representation of *all* $x \in \mathbb{R}^d$.

Theorem G.7 (Computing Kashin Representations [47])**.** *There exists a $S = O(1)$-Kashin matrix with $D = O(d)$ and there is an algorithm to obtain Kashin representations in $O(d^2 \log d)$ time.*

Chen et al. integrate Kashin representations with the PBM. We reproduce the pseudocode for the randomizer (Algorithm 2) and the analyst (Algorithm 3).

Algorithm 2: PBM Pre-processor $\mathsf{PRE}_{\theta,\ell;S,U}$ for unit ball data

Public Parameters: $\theta \in (0, 1/2)$, $\ell \in \mathbb{N}$; $S > 0$, $U \in \mathbb{R}^{d \times D}$
Input: Client data $X_i \in \mathcal{B}^d$
Output: Message vector $Y_i \in \{0, \ldots, \ell\}^D$
Compute $W_i \in \mathbb{R}^D$ by running the Kashin rep. alg. by Lyubarskii & Vershynin on X_i and U
$c \leftarrow S/\sqrt{D}$
Clamp the entries of W_i to $[-c, +c]$ `/* Not run if U is Kashin */`
$Y_i \leftarrow (\mathsf{PRE}_{c,\theta,\ell}(W_i[1]), \ldots, \mathsf{PRE}_{c,\theta,\ell}(W_i[D]))$
Return Y_i

Algorithm 3: PBM Post-processor $\mathsf{POST}_{\theta,\ell;S,U}$ for unit ball data

Public Parameters: $\theta \in (0, 1/2)$, $\ell \in \mathbb{N}$; $S > 0$, $U \in \mathbb{R}^{d\times D}$
Input: Aggregated message vector $y_{\text{agg}} \in \{0,\dots,n\ell\}^D$
Output: An estimate of the mean $\tilde{\mu}_2$
$c \leftarrow S/\sqrt{D}$
$\tilde{\mu}_\infty \leftarrow (\mathsf{POST}_{c,\theta,\ell}(y_{\text{agg}[1]}),\dots,\mathsf{POST}_{c,\theta,\ell}(y_{\text{agg}[D]}))$
$\tilde{\mu}_2 \leftarrow U\tilde{\mu}_\infty$
Return $\tilde{\mu}_2$

Claim G.8. *For any data $X \in (B_2^d)^n$, let $\mu_2 = \frac{1}{n}\sum_{i=1}^n X_i$. Let U be a S-Kashin matrix. For any $\mathbf{P}$, if $\tilde{\mu}_2 \leftarrow (\mathsf{POST}_{\theta,\ell;K,U} \circ \Sigma_{[\ell]} \circ M_{\mathbf{P}})(\vec{x})$ then*

$$\mathbb{E}\left[\left\|\tilde{\mu}_2 - \mu_2\right\|_2^2\right] \le S^2 \cdot \left(\frac{t^2}{n^2\theta^2}\left(\frac{1}{2}+\theta\right)^2 + \frac{n-t}{n}\cdot\frac{c^2}{4\ell n\theta^2}\right).$$

Proof. Define $\mu_\infty := \frac{1}{n}\sum_{i=1}^n W_i$. We first fix any realization of $\tilde{\mu}_2$, which also fixes $\tilde{\mu}_\infty$. In their work, Chen et al. showed that $\left\|\tilde{\mu}_2 - \mu_2\right\|_2^2 \le \left\|\tilde{\mu}_\infty - \mu_\infty\right\|_2^2$.

Now we factor in the randomness of the protocol (and attack). Taking expectations on either side,

$$\begin{aligned}
\mathbb{E}\left[\left\|\tilde{\mu}_2 - \mu_2\right\|_2^2\right] &\le \mathbb{E}\left[\left\|\tilde{\mu}_\infty - \mu_\infty\right\|_2^2\right] \\
&\le D\cdot c^2 \cdot \left(\frac{t^2}{n^2\theta^2}\left(\frac{1}{2}+\theta\right)^2 + \frac{n-t}{n}\cdot\frac{1}{4\ell n\theta^2}\right) && \text{(Claim G.4)} \\
&= S^2 \cdot \left(\frac{t^2}{n^2\theta^2}\left(\frac{1}{2}+\theta\right)^2 + \frac{n-t}{n}\cdot\frac{1}{4\ell n\theta^2}\right) && \square
\end{aligned}$$

If we re-scale $\ell\theta^2$ according to Theorem G.5, then we arrive at the following:

Theorem G.9. *For any ε, α and attack K, there exist parameter choices such that $\Sigma_{\in[0,\ell]} \circ \mathsf{PRE}^K_{\theta,\ell;S,U}(X)$ satisfies $(\alpha,\ \varepsilon\cdot\frac{n}{n-t})$-RDP. For any input $X \in (\mathcal{B}^d)^n$, if $\mu \leftarrow \frac{1}{n}\sum X_i$ and $\tilde{\mu} \leftarrow \Pi^K_{\theta,\ell;S,U}(X)$ then*

$$\mathbb{E}\left[\left\|\tilde{\mu} - \mu\right\|^2\right] = O\left(\frac{t^2}{n^2}\cdot\frac{\alpha\ell d}{\varepsilon n} + \frac{n-t}{n}\cdot\frac{\alpha d}{\varepsilon n^2}\right)$$

We note that the protocol and its guarantees hinges on knowing U and the values of S, D. Chen et al., like Lyubarskii & Vershynin before them, only derive asymptotic bounds. In Appendix I, we obtain a $S = 100$-Kashin matrix with overwhelming probability. We leave improvements to the constant for future work.

G.2.2 The Rotation Method

Here, we describe an approach to transform vectors such that the new ℓ_∞ norm is $O(\sqrt{\log(nd)/d})$. Although this is asymptotically worse than what is possible with Kashin representations, we show that the client-side operations can be reduced to $O(d)$ time. And for a wide range of parameters, $\sqrt{\log(nd)}$ is smaller than the Kashin constant we were able to obtain.

The idea is for clients to simply rotate their data in the same direction, then privately compute the mean of the rotated points, and finally rotate back. Rotation does not distort distances and we show that the rotated vectors all likely have ℓ_∞ norm bounded by $O(\sqrt{\log(nd)/d})$. This follows from a tail bound and a union bound over all users and coordinates.

Definition G.10. A matrix $U \in \mathbb{R}^{d\times d}$ is S-good for input $X \in (\mathcal{B}^d)^n$ if, for all X_i, $\|UX_i\|_\infty \le S/\sqrt{d}$.

Claim G.11. *Suppose $d > 16\ln(200n)$. If $U \in \mathbb{R}^{d\times d}$ is a uniformly random rotation matrix, then with probability $\ge 99/100$ it is S-good for any fixed $X \in (\mathcal{B}_2^d)^n$, where $S = \sqrt{4\ln(200nd) + 4\sqrt{\ln(200nd)} + 2}$.*

Proof. Consider any user i. $W_i \leftarrow UX_i$ is distributed as a uniformly random vector of norm $\|X_i\|_2$. If $\eta[1],\dots,\eta[d]$ are standard normal random variables, the j-th coordinate of W_i has the same distribution as $\|X_i\|_2 \cdot \frac{\eta[j]}{\sqrt{\sum_{j'\in[d]}\eta[j']^2}}$. Put differently, $W_i[j]^2$ is equal in distribution to $\|X_i\|_2^2 \cdot \frac{\eta[j]^2}{\sum_{j'\in[d]}\eta[j']^2}$ where the numerator and denominator are chi-squared random variables.

We will make use of the following technical lemma regarding chi-squared random variables:

Lemma G.12 (Laurent-Massart)**.** *If $u \sim \chi^2_d$ then*

$$\mathbb{P}\left[u > d + 2\sqrt{d\Delta} + 2\Delta\right] \le \exp(-\Delta)$$
$$\mathbb{P}\left[u \le d - 2\sqrt{d\Delta}\right] \le \exp(-\Delta)$$

By Lemma G.12 and union bounds, we know that

$$\max_j \eta[j]^2 < 1 + 2\sqrt{\Delta} + 2\Delta$$
$$\text{and} \sum_{j\in[d]} \eta[j]^2 > d/2$$

except with probability $\le d\exp(-\Delta) + \exp(-d/16)$.

By a union bound over all $i \in [n]$, the following holds except with probability $\le n(d\exp(-\Delta)+\exp(-d/16))$:

$$\max_{i,j} W_i[j]^2 < \frac{4\Delta + 4\sqrt{\Delta} + 2}{d}$$

For $\Delta = \ln(200nd)$, this tail bound is equal to S. Our lower bound on d implies a failure probability of $\le 1/100$. □

We are now ready to present the three main components of the rotation-based PBM $\overline{\Pi}_{\theta,\ell;S,U}$. The pre-processing algorithm is $\overline{\mathsf{PRE}}_{\theta,\ell;S,U}$, whose pseudocode is virtually identical to Algorithm 2 except that W_i is obtained by $W_i \leftarrow UX_i$. The predicate $\in [0,\ell]$ is the same as before: it simply checks if the given value is an integer between 0 and ℓ. The analyst's post-processing algorithm $\overline{\mathsf{POST}}_{\theta,\ell;S,U}$ is likewise identical to Algorithm 3 except that we simply revert the transformation that the clients performed ($\tilde{\mu}_2 \leftarrow U^T\tilde{\mu}_\infty$).

The following claim is immediate from substituting our value of c into Claim G.4.

Claim G.13. *Suppose U is the matrix in Claim G.11. For any attack K, if $\tilde{\mu}_2 \leftarrow \overline{\Pi}^K_{\theta,\ell;S,U}(X)$, then*

$$\mathbb{E}\left[\left\|\tilde{\mu}_2 - \mu_2\right\|_2^2\right] \le \left(4\ln(200nd) + 4\sqrt{\ln(200nd)} + 2\right)\cdot\left(\frac{t^2}{n^2\theta^2}\left(\frac{1}{2}+\theta\right)^2 + \frac{1}{\ell n^2\theta^2}\cdot\frac{n-t}{4}\right)$$

Finally, we express the expected squared-error in terms of target privacy parameters. Identical to the Kashin-based protocol, this is achieved by combining Claim G.13 with the privacy analysis in the scalar case (Claim G.3) and RDP composition (Theorem G.5).

Theorem G.14. *For any ε, α, there exist choices for $\ell \in \mathbb{N}$ and $\theta \in (0, 1/2)$ such that (a) the algorithm $\overline{Pi}^K_{\theta,\ell;S,U}$ satisfies $(\alpha,\ \varepsilon\cdot\frac{n}{n-t})$-RDP and (b) for any input $X \in (\mathcal{B}^d)^n$ and attack K, if $\tilde{\mu} \leftarrow \overline{\Pi}^K_{\theta,\ell;S,U}(X)$ then*

$$\mathbb{E}\left[\left\|\tilde{\mu} - \mu\right\|^2\right] = O\left(\log(nd)\cdot\left(\frac{t^2}{n^2}\cdot\frac{\alpha\ell d}{\varepsilon n} + \frac{n-t}{n}\cdot\frac{\alpha d}{\varepsilon n^2}\right)\right)$$

where $\mu \leftarrow \frac{1}{n}\sum X_i$.

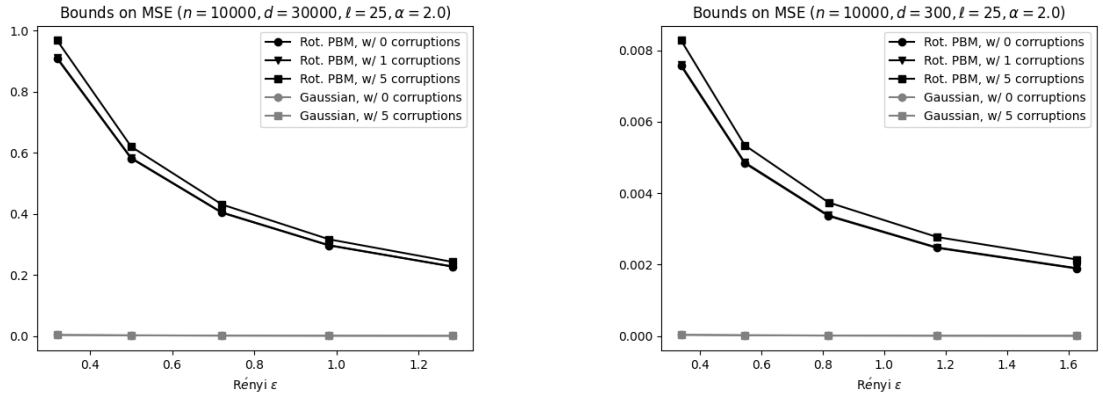

Figure 7: Visualization of our bound on mean-squared error of the rotation-based PBM as a function of Rényi ε.

Visualizing our Bounds In Figure 7, we plot the outcome of Theorem G.14 as a function of the Rényi privacy parameter ε. This is done by fixing $\ell = 25$, varying θ, and computing a bound on ε (via Theorem G.2). Naively doing so involves divisions between small numbers; the next Appendix sketches the technique we used to avoid such instability.

Due to the different privacy guarantees, we cannot immediately compare the privacy-accuracy tradeoff curve of the PBM (Figure 7 with the curve for the DBM (Figure 6). So we transform RDP ε into an approximate DP ε by applying a result by Mironov (Proposition 3 in the work that introduced RDP [49]). As shown in Figure 8, the resulting curve for the PBM (gray) is worse than the existing curve for our DBM (black).

Lemma A.1 in Chen et al.'s work could yield tighter bounds on ε after conversion. But it requires the computation of $\varepsilon(\alpha)$, the Rényi DP parameter as a function of the order α. As previously stated, Chen et al.'s closed-form bound does not have a concrete leading constant. We can numerically bound $\varepsilon(\alpha)$ for any given α but, aside from instability issues, we will need a bound that holds for *every* α.

Optimizing Running Time As written, $R_{\theta,\ell;U}$ takes $O(d^2)$ time. However, this can be improved to $O(d)$ time if we change coordinate systems.

Instead of accessing a rotation matrix U, suppose clients accessed a shared random vector $r \in \mathbb{R}^d$ whose coordinates are i.i.d. standard Gaussians. Note that the d angles $\left\{a[j] = \sin^{-1} \frac{r[j]}{\|r\|_2}\right\}_{j\in[d]}$ uniquely determine the uniformly random unit vector $\frac{r}{\|r\|_2}$ and vice versa. We can interpret $\frac{r}{\|r\|_2} \leftrightarrow a$ as a uniformly random rotation of the basis vector $(1, 0, \ldots, 0) \leftrightarrow (\pi/2, 0, \ldots, 0)$.

A client can perform the same Cartesian-to-angular transformation to their data vector. They compute $a - (\pi/2, 0, \ldots 0)$ and add it to their angular representation, before converting back to Cartesian coordinates. There are a constant number of conversions and each takes $O(d)$ time.

H Stably Computing the Privacy Parameter of PBM

In this Appendix, we derive a bound on $\varepsilon(\alpha, n)$ in the case where $\alpha = 2$ that permits stable computation. We first review the analysis for the general case given by Chen et al.: for any neighboring inputs X, X', the

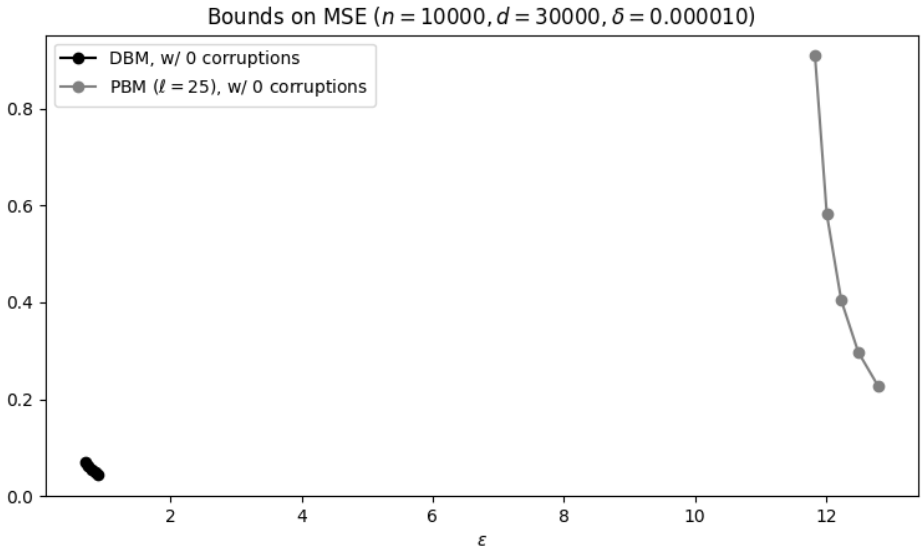

Figure 8: Comparison of error bounds for Poisson-Binomial and Distributed Binomial mechanisms under approximate DP. Privacy parameter for PBM obtained via RDP-to-approximate-DP conversion in [49].

Rényi divergence of order $\alpha > 1$ between $(\Sigma_{\in[0,\ell]} \circ \mathsf{PRE}_{c,\theta,\ell})(X)$ and $(\Sigma_{\in[0,\ell]} \circ \mathsf{PRE}^{n}_{c,\theta,\ell})(X')$ is bounded by

$$\ell \cdot \max\Big[D_\alpha\Big(\underbrace{\mathbf{Ber}(\tfrac{1}{2}-\theta) * \mathbf{Bin}(n', \tfrac{1}{2}+\theta)}_{P_1} \;\|\; \underbrace{\mathbf{Bin}(n'+1, \tfrac{1}{2}+\theta)}_{Q_1}\Big),$$
$$D_\alpha\Big(\underbrace{\mathbf{Bin}(n'+1, \tfrac{1}{2}-\theta)}_{P_2} \;\|\; \underbrace{\mathbf{Ber}(\tfrac{1}{2}+\theta) * \mathbf{Bin}(n', \tfrac{1}{2}-\theta)}_{Q_2}\Big)\Big]$$

where $n' = \lceil (n-1)/2 \rceil$. Thus, computing $\varepsilon(\alpha, n)$ can be done by evaluating the above expression.

Naively evaluating the Rényi divergence between P, Q involves computing the mass function ratio $P(v)/Q(v)$ for every point v. Because $Q(v)$ can be very small, this approach is generally unstable. In the case where $\alpha = 2$, we can express the Rényi divergence in a way that avoids this ratio.

For any value v and $k \in \{1, 2\}$, let $P_k(v), Q_k(v)$ be the probability masses placed on v by P_k, Q_k. We express the order-2 divergences in these terms:

$$\begin{aligned}
&D_{\alpha=2}(P_k \| Q_k) \\
&= \frac{1}{2-1} \ln \mathop{\mathbb{E}}_{v \sim Q_k}\left[\left(\frac{P_k(v)}{Q_k(v)}\right)^2\right] \\
&= \ln \sum_v \frac{P_k(v)^2}{Q_k(v)} \\
&= \ln \mathop{\mathbb{E}}_{v \sim P_k}\left[\frac{P_k(v)}{Q_k(v)}\right]
\end{aligned}$$

We now expand the ratio between probability masses for $k = 1$:

$$\begin{aligned}
&\frac{P_1(v)}{Q_1(v)} \\
&= \frac{\left(\frac{1}{2}-\theta\right)\cdot \mathbb{P}\left[\mathbf{Bin}(n', \frac{1}{2}+\theta) = v-1\right] + \left(\frac{1}{2}+\theta\right)\cdot \mathbb{P}\left[\mathbf{Bin}(n', \frac{1}{2}+\theta) = v\right]}{\mathbb{P}\left[\mathbf{Bin}\left(n'+1, \frac{1}{2}+\theta\right) = v\right]} \\
&= \frac{\left(\frac{1}{2}-\theta\right)\cdot \frac{n'!}{(v-1)!(n'-v+1)!}\left(\frac{1}{2}+\theta\right)^{v-1}\cdot\left(\frac{1}{2}-\theta\right)^{n'-v+1} + \left(\frac{1}{2}+\theta\right)\cdot\frac{n'!}{v!(n'-v)!}\left(\frac{1}{2}+\theta\right)^{v}\cdot\left(\frac{1}{2}-\theta\right)^{n'-v}}{\frac{(n'+1)!}{v!(n'+1-v)!}\cdot\left(\frac{1}{2}+\theta\right)^{v}\cdot\left(\frac{1}{2}-\theta\right)^{n'+1-v}} \\
&= \frac{v\cdot\left(\frac{1}{2}+\theta\right)^{v-1}\cdot\left(\frac{1}{2}-\theta\right)^{n'-v+2} + (n'-v+1)\cdot\left(\frac{1}{2}+\theta\right)^{v+1}\cdot\left(\frac{1}{2}-\theta\right)^{n'-v}}{(n'+1)\cdot\left(\frac{1}{2}+\theta\right)^{v}\cdot\left(\frac{1}{2}-\theta\right)^{n'+1-v}} \\
&= \frac{v\cdot\left(\frac{1}{2}-\theta\right)^{2} + (n'-v+1)\cdot\left(\frac{1}{2}+\theta\right)^{2}}{(n'+1)\cdot\left(\frac{1}{2}+\theta\right)\cdot\left(\frac{1}{2}-\theta\right)} \\
&= \frac{1}{(n'+1)(1/4-\theta^2)}\left((n'+1)\cdot\left(\frac{1}{2}+\theta\right)^{2} + v\left(\left(\frac{1}{2}-\theta\right)^{2} - \left(\frac{1}{2}+\theta\right)^{2}\right)\right)
\end{aligned}$$

Therefore

$$\begin{aligned}
&\mathop{\mathbb{E}}_{v\sim P_1}\left[\frac{P_1(v)}{Q_1(v)}\right] \\
&= \frac{1}{(n'+1)(1/4-\theta^2)}\left((n'+1)\cdot\left(\frac{1}{2}+\theta\right)^{2} - 2\theta \mathop{\mathbb{E}}_{v\sim P_1}[v]\right) \\
&= \frac{1}{(n'+1)(1/4-\theta^2)}\left((n'+1)\cdot\left(\frac{1}{2}+\theta\right)^{2} - 2\theta\left(\frac{n'+1}{2} + (n'-1)\theta\right)\right) \\
&= \frac{1}{(n'+1)(1/4-\theta^2)}\left((n'+1)\cdot\left(\frac{1}{2}+\theta\right)^{2} - (\theta+2\theta^2)n' - \theta + 2\theta^2\right) \\
&= \frac{1}{(n'+1)(1/4-\theta^2)}\left((1/4+\theta+\theta^2)n' + (1/4+\theta+\theta^2) - (\theta+2\theta^2)n' - \theta + 2\theta^2\right) \\
&= \frac{1}{(n'+1)(1/4-\theta^2)}\left((1/4-\theta^2)n' + 1/4 + 3\theta^2\right) \\
&= 1 + \frac{4\theta^2}{(n'+1)(1/4-\theta^2)}
\end{aligned}$$

so that $D_2(P_1\|Q_1) \le \frac{4\theta^2}{(n'+1)(1/4-\theta^2)}$

For $k = 2$,

$$\begin{aligned}
&\frac{P_2(v)}{Q_2(v)} \\
&= \frac{\mathbb{P}\left[\mathbf{Bin}(n'+1, \frac{1}{2}-\theta) = v\right]}{\left(\frac{1}{2}+\theta\right)\cdot\mathbb{P}\left[\mathbf{Bin}(n', \frac{1}{2}-\theta) = v-1\right] + \left(\frac{1}{2}-\theta\right)\cdot\mathbb{P}\left[\mathbf{Bin}(n', \frac{1}{2}-\theta) = v\right]} \\
&= \frac{\frac{(n'+1)!}{v!(n'+1-v)!}\cdot\left(\frac{1}{2}-\theta\right)^{v}\cdot\left(\frac{1}{2}+\theta\right)^{n'+1-v}}{\left(\frac{1}{2}+\theta\right)\cdot\frac{n'!}{(v-1)!(n'-v+1)!}\cdot\left(\frac{1}{2}-\theta\right)^{v-1}\cdot\left(\frac{1}{2}+\theta\right)^{n'-v+1} + \left(\frac{1}{2}-\theta\right)\cdot\frac{n'!}{v!(n'-v)!}\cdot\left(\frac{1}{2}-\theta\right)^{v}\cdot\left(\frac{1}{2}+\theta\right)^{n'-v}} \\
&= \frac{(n'+1)\cdot\left(\frac{1}{2}-\theta\right)\cdot\left(\frac{1}{2}+\theta\right)}{v\cdot\left(\frac{1}{2}+\theta\right)^{2} + (n'+1-v)\cdot\left(\frac{1}{2}-\theta\right)^{2}} \\
&= \frac{(n'+1)\cdot\left(\frac{1}{4}-\theta^2\right)}{2\theta v + (n'+1)\cdot\left(\frac{1}{4}-\theta+\theta^2\right)}
\end{aligned}$$

which means $D_2(P_2\|Q_2) = \ln \mathop{\mathbb{E}}_{v\sim P_2}\left[\frac{(n'+1)\cdot\left(\frac{1}{4}-\theta^2\right)}{2\theta v+(n'+1)\cdot\left(\frac{1}{4}-\theta+\theta^2\right)}\right]$

I Computing Kashin Representations

Here, we revisit the analysis of Lyubarskii & Vershynin in order to obtain concrete parameters of a Kashin matrix. We first re-state their notion of *uncertainty principle*.

Definition I.1 (Uncertainty Principle [47])**.** A matrix $U \in \mathbb{R}^{d\times D}$ satisfies the (η, δ)-uncertainty principle if, for all $x \in \mathbb{R}^D$ with $\leq \delta D$ nonzero entries, $\|Ux\|_2 \leq \eta \|x\|_2$

The authors of [47] prove that this property is a sufficient condition for a Kashin matrix:

Lemma I.2 (Implicit in [47])**.** *Given a matrix $U \in \mathbb{R}^{d\times D}$ that satisfies the (η, δ)-uncertainty principle, there exists an $O(\log d)$-round algorithm to generate a $S = 2(1-\eta)^{-1}\delta^{-1/2}$-Kashin representation of any x, where each round takes $O(d \cdot D)$ time.*

We now give an algorithm to generate a matrix that satisfies the uncertainty principle:

Lemma I.3 (Satisfiability of Uncertainty Principle)**.** *Fix even integer d and define $D := 4d$. Let U be the matrix formed by removing the bottom $D-d$ rows of a uniformly random $D \times D$ rotation matrix. Except with probability $\exp(-0.047d)$, U satisfies the $(\eta = 4/5, \delta = 1/100)$-uncertainty principle.*

Proof. This analysis is a concrete version of the sketch originally given by Lyubarskii & Vershynin [47]. The intuition is to combine the Johnson-Lindenstrauss lemma with an epsilon-net argument, where the net is drawn over the δ-sparse vectors.

Let $S \subset \mathbb{R}^D$ be those unit vectors where the number of nonzero entries is $\leq \delta D$. Let $\mathcal{N}$ be a maximal set of points in S such that every pair of points in $\mathcal{N}$ is at least ε from one another in ℓ_2 distance. Lyubarskii & Vershynin show that $|\mathcal{N}| \leq \left(\frac{3e}{\varepsilon\delta}\right)^{\delta D}$ [47].

We invoke a variant of the Johnson-Lindenstrauss lemma.

Claim I.4. *Sample U as in Lemma I.3. For any set of unit vectors $\mathcal{N} \subset \mathbb{R}^D$ and real value $s > 0$,*

$$\mathop{\mathbb{P}}_{U}\left[\exists x \in \mathcal{N} \quad \|Ux\|_2^2 > \frac{e^s - e^{s/2}}{e^s - 1}\right] < |\mathcal{N}| \cdot \exp((-D/2 + d - 1)\cdot s/2)$$

We prove this statement shortly. For now, substitution implies that the following likely occurs:

$$\forall x \in \mathcal{N} \ \|Ux\|_2^2 < \frac{e^s - e^{s/2}}{e^s - 1}$$

Now, fix any $y \in S$. Because $\mathcal{N}$ is maximal, there is some $x \in \mathcal{N}$ where $\left\|y - x\right\|_2 \leq \varepsilon$. Hence, the triangle inequality implies

$$\forall y \in S \ \left\|Uy\right\|_2 < \sqrt{\frac{e^s - e^{s/2}}{e^s - 1}} + \varepsilon$$

Meanwhile, the failure probability is

$$\begin{aligned}
&|\mathcal{N}| \cdot \exp((-D/2 + d - 1) \cdot s/2) \\
&\leq \left(\frac{3e}{\varepsilon\delta}\right)^{\delta D} \cdot \exp((-D/2 + d - 1) \cdot s/2) \\
&= \exp\left(\delta D \ln \frac{3e}{\varepsilon\delta} + (-D/2 + d - 1) \cdot s/2\right)
\end{aligned}$$

We choose $s = 1$ and $\varepsilon = 1/100$. It is easy to verify that the norm bound is

$$\sqrt{\frac{e - e^{1/2}}{e - 1}} + 1/100 < 4/5 = \eta$$

and the failure probability is

$$\begin{aligned}
&\exp\left(4\delta d \ln \frac{300e}{\delta} + (-d - 1)/2\right) \\
&< \exp\left(d(4\delta \ln \frac{300e}{\delta} - 1/2)\right) \\
&= \exp\left(d(\frac{1}{25} \ln 30000e - 1/2)\right) \\
&< \exp(-0.047d)
\end{aligned}$$

which concludes the proof. □

We now prove the Johnson-Lindenstrauss variant.

Proof of Claim I.4. Let $\Delta = D - d$. By the construction of U, Ux is identically distributed with the vector formed by selecting the first d coordinates of a random unit vector $v \in \mathbb{R}^D$. Hence, for all $t \in (0, 1)$,

$$\mathop{\mathbb{P}}_{U}\left[\|Ux\|_2^2 > t\right] = \mathop{\mathbb{P}}_{v}\left[\sum_{i=1}^{d} v_i^2 > t\right] \tag{14}$$

But a random unit vector $v \in \mathbb{R}^D$ is itself obtained by normalizing a spherical Gaussian: if the entries of $\eta = (\eta_1, \ldots, \eta_D)$ are normally distributed with mean 0 and variance 1, then $v_i = \eta_i / \left\|\eta\right\|_2$. Hence,

$$\begin{aligned}
\mathop{\mathbb{P}}_{v}\left[\sum_{i=1}^{d} v_i^2 > t\right] &= \mathop{\mathbb{P}}_{\eta}\left[\frac{\sum_{i=1}^{d} \eta_i^2}{\sum_{j=1}^{D} \eta_j^2} > t\right] \\
&= \mathop{\mathbb{P}}_{a \sim \chi^2(d), b \sim \chi^2(\Delta)}\left[\frac{a}{a+b} > t\right] && (15) \\
&= \mathop{\mathbb{P}}_{c \sim \beta(d/2, \Delta/2)}\left[c > t\right] && (16) \\
&= 1 - I_t(d/2, \Delta/2) && \text{(CDF of } \beta\text{)} \\
&= I_{1-t}(\Delta/2, d/2)
\end{aligned}$$

(15) comes from the definition of the chi-square distribution. Meanwhile, (16) comes from the definition of the beta distribution. Now, we invoke the identity

$$\mathop{\mathbb{P}}_{c' \sim \mathbf{Bin}(n,p)} [c' \geq k] = I_p(k, n-k+1)$$

for the parameter values $p = 1-t$, $k = \Delta/2$, $n = D/2-1$ to conclude

$$\mathop{\mathbb{P}}_{U} \left[\|Ux\|_2^2 > t \right] = \mathop{\mathbb{P}}_{c' \sim \mathbf{Bin}(D/2-1, 1-t)} [c' > \Delta/2]$$

Our claim will now come from a Chernoff bound. For any $s > 0$,

$$\begin{aligned}
&\mathop{\mathbb{P}}_{c' \sim \mathbf{Bin}(D/2-1,1-t)} [c' > \Delta/2] \\
&\leq \exp(-\Delta s/2) \cdot \mathop{\mathbb{E}}_{c' \sim \mathbf{Bin}(D/2-1,1-t)} [\exp(sc')] && \text{(Markov's)} \\
&= \exp(-\Delta s/2) \cdot \mathop{\mathbb{E}}_{c''_1,\ldots,c''_{D/2-1}} \left[\exp(s \sum_{j=1}^{D/2-1} c''_j) \right] && \text{(if } c''_j \sim \mathbf{Ber}(1-t)\text{)} \\
&= \exp(-\Delta s/2) \cdot \prod_{j=1}^{D/2-1} \mathop{\mathbb{E}}_{c''_j} \left[\exp(sc''_j) \right] && \text{(Independence)} \\
&= \exp(-\Delta s/2) \cdot (t + (1-t)e^s)^{D/2-1} && \text{(MGF of Bernoulli)} \\
&= \exp(-\Delta s/2 + (D/2-1)s/2) && (17)
\end{aligned}$$

We obtain the last step by solving $t + (1-t)e^s = e^{s/2}$ for t. Substituting $\Delta = D - d$ and a union bound over $\mathcal{N}$ completes the proof. □

J Proof Sketch of Theorem 4.2.

The description of the protocol is presented in Figure 9 and the proof sketch of Theorem 4.2 is given below.

Let $\mathcal{A}$ be the static adversary who maliciously corrupts parties in protocol Π_{dCP}. Let $\mathcal{C}$ be the subset of verifiers corrupted by $\mathcal{A}$ where $|\mathcal{C}| \leq t$.

We now describe the simulator Sim. Sim internally invokes the adversary $\mathcal{A}$ with some auxiliary input z. We consider two cases depending on whether $\mathcal{A}$ corrupts the prover or not.

Simulating the case where the Prover is not corrupted. We first describe how to simulate the Commit Phase. Whenever an honest prover $\mathcal{P}$ commits to an unknown value w, Sim receives a message $(\text{receipt}, sid, \mathcal{P})$. Upon receiving the receipt message, Sim generates a commitment $\widetilde{\mathsf{com}}$ via zkMVPC's simulator Sim′ i.e., $(\widetilde{\mathsf{com}}, \mathsf{pp}, \mathsf{trap}) \leftarrow \mathsf{Sim}'(1^\lambda, t)$ and broadcasts a commitment $\widetilde{\mathsf{com}}$ on behalf of honest $\mathcal{P}$.

Next, the simulation of the Prove Phase proceeds as follows. Whenever an honest $\mathcal{P}$ sends the $(\text{dCP-prover}, sid, x_j, \mathcal{V}_j)$ message to $\mathcal{F}_{\mathsf{dCP}}$, Sim receives the message $(\text{dCP-proof}, sid, x_j, \mathsf{accept})$ from $\mathcal{F}_{\mathsf{dCP}}$. If $\mathcal{V}_j \in \mathcal{C}$, then Sim internally sends the message $(x_i, \widetilde{\pi}_j)$ to $\mathcal{A}$ on behalf of $\mathcal{P}$ where $\widetilde{\pi}_j$ is the simulated proof which can be simulated using the same simulator as the one used for one-to-many argument system in [68].

Simulating the case where the Prover is corrupted. The simulation of Commit Phase is as follows. Whenever $\mathcal{A}$ (controlling $\mathcal{P}$) wants to commit to a value, Sim obtains the commitment $\mathsf{com}_{\mathbb{V}_d}$ that $\mathcal{A}$ broadcasts to all verifiers. Then, Sim extracts the input $\mathbf{in}$ from the commitment and externally sends the message $(\mathsf{commit}, \mathsf{sid}, \mathcal{P}, w)$ to $\mathcal{F}_{\mathsf{dCP}}$ where $w = \mathbf{in}$. Also, Sim records the value w.

Next, the Prove Phase is simulated as follows. Whenever $\mathcal{A}$ want to prove a statement to a verifier $\mathcal{V}_j$, Sim receives the message $(\mathbf{out}_j, \pi_j)$ on behalf of the honest verifier $\mathcal{V}_j$. Then, Sim first verifiers the proof π_j (as per

Protocol Π_{dCP}

Protocol Π_{dCP} runs between the Prover $\mathcal{P}$ and n Verifiers $\mathcal{V} = \{\mathcal{V}_1, \ldots, \mathcal{V}_n\}$. It is parameterized by n relations $(\mathcal{R}_1, \ldots, \mathcal{R}_n)$, which are specified via circuit C as per Notation 4.1. Given a $\mathbb{d}$-depth layered arithmetic circuit $\mathsf{C} : \mathbb{F}^{t+1} \to \mathbb{F}^n$ associated with relations $(\mathcal{R}_1, \ldots, \mathcal{R}_n)$, the prover $\mathcal{P}$ needs to convince $\mathcal{V}_j$ that $\mathbf{out}_j = [\mathsf{C}(\mathbf{in})]_j \neq \perp$, where $[C(\mathbf{in})]_j$ is the j-th output of the circuit given input $\mathbf{in}$, and $\mathbf{out}_j$ is the claimed result for $\mathcal{V}_j$. In other words, $\mathcal{P}$ needs to $\mathcal{V}_j$ that $(\mathbf{out}_j, \mathbf{in}) \in \mathcal{R}_j$.
Without loss of generality, assume that the lengths of the input and output to C are both powers of 2, and we can pad them if not. Let ρ be a random oracle and λ be the security parameter.

Commit Phase. Set $\mathsf{pp} \leftarrow \mathsf{zkMVPC.KeyGen}(1^\lambda, t)$. $\mathcal{P}$ invokes $\mathsf{zkMVPC.Commit}(\tilde{\mathbb{V}}_d, \mathsf{pp})$ to generate $\mathsf{com}_{\tilde{\mathbb{V}}_d}$ and broadcasts $\mathsf{com}_{\tilde{\mathbb{V}}_d}$ to all verifiers. $\tilde{\mathbb{V}}_d$ is the multilinear extension of $\mathbf{in}$, as defined in the GKR protocol.

Prove Phase. The Open phase is as follows.

1. For each $j \in [n]$, $\mathcal{P}$ sets $\mathbb{V}_0(j)$ as $\mathbf{out}_j$ for all $i \in [n]$.
2. For each $j \in [n]$, $\mathcal{P}$ runs the sumcheck protocol on:

$$\tilde{\mathbb{V}}_0(\tilde{j}) = \sum_{\vec{x} \in \{0,1\}^{\log n}} \tilde{\beta}(\vec{j}, \vec{x}) \tilde{\mathbb{V}}_0(\vec{x})$$

   where $\vec{j}$ is the binary string of j. For $i = 1, \ldots, \log n$:

   - Suppose $M_{i,j}$ is the i-th univariate polynomial $\mathcal{P}$ sends to $\mathcal{V}_j$ in the sumcheck. If $i = 1$, set $r_{i,j} = \rho(\mathsf{com}_{\tilde{\mathbb{V}}_d} \| \mathbb{V}_0(j) \| M_{1,j})$. If $i > 1$, set $r_{i,j} = \rho(g_{i-1}^{(0)} \| M_{i,j})$.
   - $\mathcal{P}$ builds a Merkle tree on the vector $\vec{r}^{(i)} = (r_{i,0}, \ldots, r_{i,N-1})$. Let $g_i^{(0)} = \mathsf{MT.Commit}(\vec{r}^{(i)})$. Then $\mathcal{P}$ assigns $g_i^{(0)}$ as the common random challenge in the i-th round.
   - $\mathcal{P}$ attaches $(r_{i,j}, \mathsf{path}_{i,j}) \leftarrow \mathsf{MT.Open}(j, g_i^{(0)})$ in the proof associated with verifier $\mathcal{V}_j$ i.e., π_j.

   In the last round of the sumcheck, $\mathcal{P}$ attached $\tilde{\mathbb{V}}_0(\vec{g}^{(0)})$ to each $\mathcal{V}_j$'s proof π_j (as the all the verifiers share the same random vector $\vec{g}^{(0)}$).
3. $\mathcal{P}$ invokes the GKR protocol with $\tilde{\mathbb{V}}_0(\vec{g}^{(0)})$. In each round, $\mathcal{P}$ generates the random challenges by querying ρ on the last round's challenge and message. For all $\mathcal{V}_j$, the random challenges and the transcript would be exactly the same because they share the same claim about $\tilde{\mathbb{V}}_0(\vec{g}^{(0)})$ and the same random vector $\vec{g}^{(0)}$ from the first round of this step.
4. In the last round of the GKR protocol, all verifiers have the same claim about $\tilde{\mathbb{V}}_d(\vec{g}^{(d)})$. $\mathcal{P}$ invokes $(y_{gkr}, \pi_{gkr}) \leftarrow \mathsf{zkMVPC.Open}(\tilde{\mathbb{V}}_d, \vec{g}^{(d)}, \mathsf{pp})$ (to generate the proof for the claim) and attaches its output (y_{gkr}, π_{gkr}) to each of the proofs π_j for all $j \in [n]$. For each $\mathcal{V}_j$, $\mathcal{P}$ sends the message $(\mathbf{out}_j, \pi_j)$ to $\mathcal{V}_j$.
5. **(Verification)** For each $j \in [n]$, $\mathcal{V}_j$ outputs $(\mathbf{out}_j, \mathsf{accept})$ if the all of the following checks pass; otherwise outputs reject.
   - checks the proof π_j with random challenges provided by $\mathcal{P}$ and $\mathsf{zkMVPC.Verify}(y_{gkr}, \pi_{gkr})$
   - checks all authenticated paths in the Merkle tree proof by $\mathsf{MT.Verify}$.[a]
   - $r_{i,j} = g_i^{(0)}$
   - queries the random oracle ρ to check the generation process of random challenges.

[a] In particular, given $\mathsf{path}_{i,j} = (v_1, \ldots, v_{\log n})$, for $k = 1, \ldots, \log n$: If $j_i = 0$, $\mathcal{V}_j$ computes $r_{i,j} = \rho(r_{i,j} \| v_i \| (i-1)(\log n + 1) + k)$; otherwise, $\mathcal{V}_j$ computes $r_{i,j} = \rho(v_i \| r_{i,j} \| (i-1)(\log n + 1) + k)$.

Figure 9: Distributed Commit and Prove Protocol

the verification steps described in Figure 9). Sim aborts if the proof verification passes but $\mathcal{R}_j(\mathbf{out}_j, w) \neq 1$. If

Sim does not abort, then Sim sets $x_j := \mathbf{out}_j$ if the proof verification passes; otherwise set x_j to some arbitrary value x'_j such that $\mathcal{R}_j(x'_j, w) \neq 1$. Finally, Sim externally sends the message (dCP-prover, $sid, x_j, \mathcal{V}_j$) to $\mathcal{F}_{\mathsf{dCP}}$. Finally, Sim outputs whatever $\mathcal{A}$ outputs and halts.

We now prove that the real world view is computationally indistinguishable from the ideal world view. When the prover is uncorrupted, the key difference between the ideal and real executions is that the commitment $\widetilde{\mathsf{com}}$ and proof $\{\widetilde{\pi}_j\}_{\mathcal{V}_j \in \mathcal{C}}$ are both simulated in the former and generated as per the protocol in the latter. It follows from the zero-knowledge property of zkMVPC and the zero-knowledge property of the one-to-many argument system[18] of [68] that the **REAL** and **IDEAL** distributions are indistinguishable (when the prover is not corrupted).

When the prover is corrupted, we first claim that Sim aborts with negligible probability. More precisely, we claim that if $\mathcal{R}_j(\mathbf{out}_j, w) \neq 1$, then the proof verification on input $(\mathbf{out}_j, \pi_j)$ corresponding to an honest verifier $\mathcal{V}_j$ fails, except with negligible probability. This follows from the soundness of the one-to-many argument system of [68].

Assuming that Sim does not abort, we next claim that the outputs of the honest verifiers are the same in both the real execution with $\mathcal{A}$ and the ideal execution with Sim. If an honest verifier $\mathcal{V}_j$ outputs (x_j, accept) in the real execution, we show that $\mathcal{V}_j$ also outputs the same. If $\mathcal{V}_j$ accepts, then the proof verification of $(\mathbf{out}_j, \pi_j)$ passed. This in turn implies $\mathcal{R}_j(x_j, w) = 1$ (as we assumed Sim does not abort), where $x_j := \mathbf{out}_j$ and w is extracted from the commitment. In the ideal execution, upon receiving $(\mathbf{out}_j, \pi_j)$ internally from $\mathcal{A}$, Sim will send (dCP-Prove, sid, $x_j, \mathcal{V}_j$) to $\mathcal{F}_{\mathsf{dCP}}$. This causes $\mathcal{V}_j$ to output (x_j, accept), same as the real execution. A similar reasoning can be used to show that an honest verifier $\mathcal{V}_j$ outputs reject in real execution, then we show that $\mathcal{V}_j$ also outputs reject in the ideal execution.

K Proof of Theorem 4.3.

Let $\mathcal{A}$ be an adversary in the real world. We show the existence of a simulator Sim such that for any set of corrupted verifiers $\mathcal{C} \subset \mathcal{V}$ and for all inputs, the **REAL** and **IDEAL** distributions are indistinguishable. We separately deal with the case where the Prover $\mathcal{P}$ is honest and the case that the Prover $\mathcal{P}$ is corrupted. Roughly, when the prover is honest, we show that the honest verifiers always accept the dealt shares, and in particular that the adversary cannot falsely generate complaints that will interfere with the result.

Case I: The Prover is Honest. In this case, in an ideal execution, the Prover sends a t-degree polynomial f to the functionality $\mathcal{F}_{\mathsf{VRS}}$ and each honest verifier $\mathcal{V}_i \in \mathcal{H}$ receives $(\mathsf{Share}_i, \mathsf{accept})$ from $\mathcal{F}_{\mathsf{VRS}}$, outputs it and never outputs reject, where $\mathsf{Share}_i := f(\alpha_i)$. Observe that the adversary cannot influence the output as none of the corrupted parties have inputs. We first show that this holds in the real execution as well i.e., in a real execution each honest verifier always output $(\mathsf{Share}_i, \mathsf{accept})$ and never outputs reject.

Since the Prover is honest, it executes the input sharing with input f as prescribed by the protocol. In this case, we show that an honest verifier always outputs $(f(\alpha_i), \mathsf{accept})$. This follows from the claim that each honest verifier $\mathcal{V}_i \in \mathcal{H}$ *keep their existing share* sh'_i (which is equal to $f(\alpha_i)$) they received from the honest prover $\mathcal{P}$ in the first round and output it (i.e., $\mathsf{Share}_i := \mathsf{sh}'_i$ as per step 4(b)i in Figure 3). Hence, it is sufficient to show that all honest verifiers execute step 4(b)i. This occurs because each honest verifier $\mathcal{V}_i \in \mathcal{H}$ satisfies the following conditions:

1. $\mathsf{happy}_i = \mathsf{accept}$: Since the prover $\mathcal{P}$ is honest, this follows ideal functionality $\mathcal{F}_{\mathsf{dCP}}$.
2. Consistency check (described in step 3 in Figure 3) passes: Since $\mathcal{P}$ is honest, the masked shares broadcasted by all honest verifiers are correctly computed and there can be at most t errors corresponding to the masked shares broadcasted on behalf of corrupted verifiers. Decoding procedure applied to the masked shares succeeds as there are at most t errors.

[18] The one-to-many argument system is detailed in Section 3 of [68].

Since the outputs of the honest verifiers are fully determined by the honest prover's input, it remains to show the existence of an ideal model adversary/simulator Sim that can generate the view of the adversary $\mathcal{A}$ in an execution of the real protocol, given only the outputs of the corrupted verifiers $\mathcal{C}$. The description of Sim is as follows:

1. Sim internally invokes the adversary $\mathcal{A}$.
2. *Interaction with the ideal functionality $\mathcal{F}_{\mathsf{VRS}}$:* Sim receives the output shares $\{\mathsf{Share}_i\}_{\mathcal{V}_i \in \mathcal{C}}$ of corrupted verifiers.
3. *Generating the view of the corrupted parties:* Sim generates the view of $\mathcal{A}$ in the offline phase by running the honest parties as per the real protocol. Next, Sim chooses a polynomial $f'(\cdot)$ such that $f'(\alpha_i) = \mathsf{Share}_i$ for all corrupted verifiers $\mathcal{V}_i \in \mathcal{C}$ and $(f'(\alpha_i), f') \in \mathcal{R}_i$. Then, Sim runs all honest parties (both prover and all honest verifiers) in an interaction with $\mathcal{A}$ where the prover's input is $f'(\cdot)$. In more detail, Sim emulates the $\mathcal{F}_{\mathsf{dCP}}$ functionality by invoking the commit and prove phases with prover's input f'.
4. Sim outputs whatever $\mathcal{A}$ outputs and aborts.

Informally, the ideal world differs from the real world in that the shares $\{\mathsf{Share}_i\}_{\mathcal{V}_i \in \mathcal{C}}$ are generated based on polynomial $f(\cdot)$ in the real world or the simulator-chosen polynomial $f'(\cdot)$ in the ideal world. The indistinguishability of the views in the real world and the ideal world follows from the security of the secret-sharing scheme. In more detail, the adversary can only get up to a threshold of shares corresponding to the inputs of the honest prover and these are distributed uniformly at random over the underlying field. We state the correctness of the simulation in the following lemma.

Lemma K.1. *The following two distribution ensembles are computationally indistinguishable,*

$$\left\{\mathbf{REAL}_{\Pi_{\mathsf{VRS}},\mathcal{A},\mathcal{C}}((p,w),n)\right\}_{n\in\mathbb{N}} \approx^{c} \left\{\mathbf{IDEAL}_{\mathcal{F}_{\mathsf{VRS}},\mathsf{Sim}(z),\mathcal{C}}((p,w),n)\right\}_{n\in\mathbb{N}}.$$

where $f(\cdot)$ is t-degree polynomial.

Towards proving this indistinguishability, we consider a sequence of intermediate hybrid experiments and apply a standard hybrid argument. For each hybrid experiment **Hybrid** H_i, we define the random variable $\mathsf{hyb}_i(n)$ that denotes the output of the experiment.

Hybrid H_0**:** This hybrid is the real world execution of the protocol Π_{VRS}. By construction, we have that $\mathsf{hyb}_0(n) \equiv \mathbf{REAL}_{\Pi_{\mathsf{VRS}},\mathcal{A},\mathcal{C}}$.

Hybrid H_1**:** This hybrid is similar to **Hybrid** H_0 with the exception that the masked input shares, denoted by $\{\widehat{\mathsf{msh}}_1, \ldots, \widehat{\mathsf{msh}}_n\}$, are randomly sampled such that these shares satisfy the constraints: (i) it forms a t-degree codeword $\mathsf{RS}_{\mathbb{F},n,t+1}$ and (ii) are consistent with the view of the adversary i.e., $\widehat{\mathsf{msh}}_i = \mathsf{Share}'_i + \mathsf{rsh}_i$ for all $\mathcal{V}_i \in \mathcal{C}$. These randomly sampled masked input shares are broadcast during step 2 of the protocol Π_{VRS}.

Lemma K.2. $\{\mathsf{hyb}_0(n)\}_{n\in\mathbb{N}}$ *and* $\{\mathsf{hyb}_1(n)\}_{n\in\mathbb{N}}$ *are identically distributed.*

Proof. Hybrids H_0 and H_1 only differ with respect to how the masked input shares are computed during the share recovery phase. In hybrid H_0, the random shares $\{\mathsf{rsh}_{i,1}, \ldots, \mathsf{rsh}_{i,n}\}$ are sampled first such that they form a codeword $\mathsf{RS}_{\mathbb{F},n,t+1}$ and then the masked input shares are computed as $\widehat{\mathsf{msh}}_i = \mathsf{Share}'_i + \mathsf{rsh}_i$, which makes the resulting masked shares uniformly distributed such that they form a codeword and are consistent with the view of the adversary. Whereas, in hybrid H_1, the masked input shares are first sampled as per the two constraints (i) form a t-degree codeword $\mathsf{RS}_{\mathbb{F},n,t+1}$ and (ii) are consistent with the view of the adversary. This fixes the random shares to be $\mathsf{rsh}_i = \widehat{\mathsf{msh}}_i - \mathsf{Share}'_i$. These two approaches of computing the masked input shares result in identical distributions, hence hybrids H_0 and H_1 are identically distributed.

□

Hybrid H_2: This hybrid is similar to the previous hybrid with the exception that the output shares of honest verifiers are computed as per the simulation where Sim chooses the t-degree polynomial $f'(\cdot)$ and witness w' that is consistent with the corrupted verifiers' shares i.e. $f'(\cdot) = \mathsf{Share}_i$ and $(f'(\alpha_i), f') \in \mathcal{R}_i$ for all $\mathcal{V}_i \in \mathcal{C}$.

Lemma K.3. *$\{\mathsf{hyb}_1(n)\}_{n\in\mathbb{N}}$ and $\{\mathsf{hyb}_2(n)\}_{n\in\mathbb{N}}$ are statistically indistinguishable.*

Proof. Note that the adversary's view is generated in the same way in both hybrids $\{\mathsf{hyb}_1(n)\}_{n\in\mathbb{N}}$ and $\{\mathsf{hyb}_2(n)\}_{n\in\mathbb{N}}$ except for the output shares.

We first claim that the output party $\mathcal{O}$ receives output shares that encode the same output Y_{agg} in both the hybrid H_1 and H_2. Specifically, in hybrid H_1, the honest verifiers aggregate the input shares of all verifiers in $\mathcal{V}$ to obtain the output shares and send these output shares to the output party. The output shares held by the honest verifiers determine the output Y_{agg}.

During the course of execution of the protocol, the adversary corrupts at most $\mathsf{n_c} - 1$ clients, t verifiers, and the output party $\mathcal{O}$. The adversary's view comprises of at most t input shares, the proof and masked input shares of each honest client. It follows from the privacy property of the packed secret-sharing scheme (based on $\mathsf{RS}_{\mathbb{F},n,t+1}$ encoding) that the adversary cannot learn any information about the inputs by honest clients from these t shares. On the other hand, the proofs and masked input shares do not reveal any information as they have been simulated independent of the honest clients' inputs. Thus, the hybrids $\{\mathsf{hyb}_1(n)\}_{n\in\mathbb{N}}$ and $\{\mathsf{hyb}_2(n)\}_{n\in\mathbb{N}}$ are indistinguishable.

□

Hybrid H_3: This hybrid is similar to the hybrid H_4 except that the masked input shares associated with honest clients are simulated as per H_2.

Lemma K.4. *$\{\mathsf{hyb}_2(n)\}_{n\in\mathbb{N}}$ and $\{\mathsf{hyb}_3(n)\}_{n\in\mathbb{N}}$ are statistically indistinguishable.*

Proof. The proof is analogous to the arguments in Lemma K.2. □

Hybrid H_4: This is an ideal execution of the protocol, where the simulator described above interacts with the adversary and the ideal functionality for $\mathcal{F}_{\mathsf{VRS}}$.

Lemma K.5. *$\{\mathsf{hyb}_3(n)\}_{n\in\mathbb{N}}$ and $\{\mathsf{hyb}_4(n)\}_{n\in\mathbb{N}}$ are statistically indistinguishable.*

Proof. The proof is analogous to the arguments in Lemma K.3.

□

Case II: The Prover is Corrupted. In this case, $\mathcal{A}$ controls the prover and corrupt verifiers $\mathcal{C}$. Roughly, Sim plays the role of all the honest verifiers. Recall that the prover is the only party with an input and each of the verifier has output that is fully determined by the prover's input. If the simulated execution is such that the verifiers output $\perp$, then the simulator Sim sends an invalid polynomial, say $f'(x) = x^{2t+2}$, to $\mathcal{F}_{\mathsf{VRS}}$ functionality. Otherwise, the simulator

The description of the simulator Sim is as follows:

1. Sim invokes $\mathcal{A}$ on its auxiliary input z.
2. Sim plays the role of all the honest verifiers interacting with $\mathcal{A}$ as per the protocol Π_{VRS}, running until the end. This includes emulating $\mathcal{F}_{\mathsf{dCP}}$ functionality with $\mathcal{A}$ (that plays the role of the prover) during commit and prove phases. Let the input sent by $\mathcal{A}$ (on behalf of $\mathcal{P}$) during the Commit Phase of $\mathcal{F}_{\mathsf{dCP}}$ be $(\mathsf{Input}, \mathsf{sid}, \mathcal{P}, \widehat{f})$. Let Share'_i be the output share of $\mathcal{V}_i \in \mathcal{V}$ at the end of Π_{VRS} protocol.

3. *Interaction with $\mathcal{F}_{\mathsf{VRS}}$:* Depending on the output of honest verifiers during the emulation of the Prove phase of $\mathcal{F}_{\mathsf{dCP}}$ functionality, Sim proceeds as follows:

   (a) If the consistency check fails, then Sim sends $(\mathsf{Input}, \mathsf{sid}, \mathcal{P}, f')$ as the prover's input to $\mathcal{F}_{\mathsf{VRS}}$ where $f'(\cdot)$ is an arbitrary polynomial of degree greater than t (say $f'(x) = x^{2t+2}$). This causes $\mathcal{F}_{\mathsf{VRS}}$ to send $(\mathsf{Output}, \mathsf{sid}, \mathcal{V}_i, \mathsf{reject})$ to all the verifiers $\mathcal{V}_i \in \mathcal{V}$ in the ideal world.

   (b) If the consistency check passes, then Sim aborts if either of the following check holds for some honest verifier $\mathcal{V}_i \in \mathcal{H}$:

      (i) $\widehat{f}(\alpha_i) \neq \mathsf{Share}'_i$

      (ii) $\left(\mathsf{Share}'_i, \widehat{f}\right) \notin \mathcal{R}_i$

      where Share'_i is computed as per step 4(b)ii in Figure 3. If Sim does not abort, then Sim sends $(\mathsf{Input}, \mathsf{sid}, \mathcal{P}, \widehat{f})$ to $\mathcal{F}_{\mathsf{VRS}}$ on behalf of the $\mathcal{P}$. This causes $\mathcal{F}_{\mathsf{VRS}}$ functionality to send the output $(\mathsf{Output}, \mathsf{sid}, \mathcal{V}_i, \widehat{f}(\alpha_i), \mathsf{accept})$ to each verifiers $\mathcal{V}_i \in \mathcal{V}$ in the ideal world.

4. Sim outputs whatever $\mathcal{A}$ outputs and halts.

First, observe that Sim plays the role of all the honest verifiers in the ideal execution as per the specification of the protocol. The honest verifiers do not have any inputs and only samples random values during the offline phase. Sim samples the random values needed for the offline phase on behalf of the honest verifiers in the ideal executions. Since the Sim follows the exact specification of protocol Π_{VRS}, the messages sent by Sim during the ideal execution are identically distributed as those sent by the honest verifiers in the real execution of protocol Π_{VRS}.

Next, we claim that the outputs of corrupt verifiers are the same in both the real and ideal executions. This follows from the fact that the output of the verifiers only depends on the prover's input (and not on the randomness sampled by the verifiers in the offline phase). Since the prover's input is determined by $\mathcal{A}$ and $\mathcal{A}$ is determinitic[19], the output of corrupt verifiers is fully determined and are the same both in the real and ideal executions.

Now, it remains to be shown that the outputs of the honest verifiers are also same in the real and ideal executions. We consider two cases.

1. **There exists an honest verifier that outputs** reject **in the real execution.** An honest verifier outputs reject if the consistency check fails i.e., the decoding of the broadcasted masked shares fails. Since all the honest verifiers receives the same broadcasted masked shares, they also output reject in the real execution. In the ideal execution, if the consistency check fails, Sim sends an invalid polynomial, say $f'(x) := x^{2t+2}$, to $\mathcal{F}_{\mathsf{VRS}}$. As a result, all the honest verifiers receive reject from $\mathcal{F}_{\mathsf{VRS}}$ and output it.

2. **There exists an honest verifier that does not output** reject **in the real execution.** We first observe that whether an honest verifier outputs accept or reject depends on the consistency check in both the real and ideal executions. Further, since the consistency check is based on values broadcast by the verifiers (i.e., broadcasted masked shares), all the verifiers make the decision. If the consistency check passes, then all the honest verifiers output accept along with their respective output shares in both real and ideal executions. It suffices to show that, in both the real and ideal executions, if an honest verifier $\mathcal{V}_i \in \mathcal{H}$ outputs $(\mathsf{Output}, \mathsf{sid}, \mathcal{V}_i, \mathsf{Share}'_i, \mathsf{accept})$, then it holds that $\mathsf{Share}'_i = \widehat{f}$. In the ideal execution, this holds because Sim sends $\widehat{f}$ as input to the $\mathcal{F}_{\mathsf{VRS}}$ functionality (since the consistency check passes). This results in honest verifiers outputting accept along with the share $\mathsf{Share}'_i = \widehat{f}(\alpha_i)$.

   Next, we show that this also holds for the real execution.

   Lemma K.6. *If an honest verifier $\mathcal{V}_i \in \mathcal{V}$ outputs $(Output, \mathsf{sid}, \mathcal{V}_i, \mathsf{Share}'_i, \mathsf{accept})$, then it holds that $\mathsf{Share}'_i = \widehat{f}(\alpha_i)$ in the real execution.*

[19] We assume $\mathcal{A}$ is deterministic as its auxiliary inputs can contain the "best" random coins for its attack.

Proof. We first show that there exists a set of at least $n - 2t > t + 1$ honest verifiers, denoted by the set $\mathcal{T}_1$, such that $\mathsf{Share}'_i = \widehat{f}(\alpha_i)$ for $\mathcal{V}_i \in \mathcal{T}_1$. Since we assumed that the consistency check passed, there must be at most t honest verifiers $\mathcal{V}_i \in \mathcal{H}$ with $\mathsf{happy}_i = \mathsf{reject}$ (otherwise, decoding of $(\mathsf{msh}'_1, \ldots, \mathsf{msh}'_n)$ would fail as more t of the masked shares would have been set to $\perp$). It follows that there exist at least $n - 2t \geq t + 1$ honest verifiers, denoted by the set $\mathcal{T}_1$, such that $\mathsf{happy}_i = \mathsf{accept}$ and hence the *keep their existing share* i.e., $\mathsf{Share}'_i := \mathsf{sh}_i$. It follows from the $\mathcal{F}_{\mathsf{dCP}}$ functionality that $\mathsf{sh}_i = \widehat{f}(\alpha_i)$. Hence, we have $\mathsf{Share}'_i = \widehat{f}(\alpha_i)$ for $\mathcal{V}_i \in \mathcal{T}_1$.

Next, we show that $\{\mathsf{Share}'_i\}_{\mathcal{V}_i \in \mathcal{H}}$ should form a valid $\mathsf{RS}_{\mathbb{F},n,t+1}$ codeword. We focus on honest verifiers with $\mathsf{happy}_i = \mathsf{reject}$ i.e., denoted by the set $\mathcal{T}_2 := \mathcal{H} \backslash \mathcal{T}_1$. For all $\mathcal{V}_i \in \mathcal{T}_2$, Share'_i is recovered as follows: $\mathsf{Share}'_i := \mathsf{msh}''_i + \mathsf{rsh}_i = (\widehat{f}(\alpha_i) + \mathsf{rsh}_i) - \mathsf{rsh}_i = \widehat{f}(\alpha_i)$. Now, it suffices to show that $\mathsf{msh}''_i = (\widehat{f}(\alpha_i) + \mathsf{rsh}_i)$ for all $\mathcal{V}_i \in \mathcal{T}_2$. For all $\mathcal{V}_i \in \mathcal{T}_1$, we have $\mathsf{msh}''_i = \mathsf{msh}'_i = \mathsf{sh}_i + \mathsf{rsh}_i = \widehat{f}(\alpha_i) + \mathsf{rsh}_i$ (follows from the protocol description). Note that $(\widehat{f}(\alpha_1), \ldots, \widehat{f}(\alpha_n))$ and $(\mathsf{rsh}_1, \ldots, \mathsf{rsh}_n)$ are $\mathsf{RS}_{\mathbb{F},n,t+1}$ codewords. Thus, by the linearity of the coding scheme and the fact that $(\mathsf{msh}''_1, \ldots, \mathsf{msh}''_n)$ is an $\mathsf{RS}_{\mathbb{F},n,t+1}$ codeword, we have that $\mathsf{msh}''_i = \widehat{f}(\alpha_i) + \mathsf{rsh}_i$ for all honest verifiers $\mathcal{V}_i \in \mathcal{H}$.

Together, we obtain that $\mathsf{Share}'_i = \widehat{f}(\alpha_i)$ for all honest verifiers $\mathcal{V}_i \in \mathcal{H}$. □

This completes the proof for the corrupt prover case.

L Input-Certified Secure Aggregation Protocol

The description of the protocol is presented in Figure 10.

M Proof of Theorem 3.15

First, we claim that the canonical protocol in the $\mathcal{F}_{\mathsf{DBM}}$-hybrid is DP-DME protocol (as per Definition 2.3). Second, we show that the protocol Π_{DBM} described above securely realizes $\mathcal{F}_{\mathsf{DBM}}$. Finally, by standard composition we can conclude that Π_{DBM} is as a DP-DME protocol.

Lemma M.1. *The canonical protocol in the $\mathcal{F}_{\mathsf{DBM}}$-hybrid (described in Figure 11) is a* DP-DME *protocol.*

Proof. As demonstrated in Theorem 3.6, $\mathcal{F}_{\mathsf{DBM}}$ adheres to the (ϵ, δ)-IND-CDP. Furthermore, according to Theorem 3.12, $\mathcal{F}_{\mathsf{DBM}}$ also fulfills the property of ν-DME-Usefulness. Thus, $\mathcal{F}_{\mathsf{DBM}}$ is a DP-DME functionality. □

Lemma M.2. *Protocol Π' in the $\mathcal{F}_{\mathsf{Agg}}$-hybrid securely realizes $\mathcal{F}_{\mathsf{DBM}}$.*

Proof. The main difference between the protocol Π' and functionality $\mathcal{F}_{\mathsf{DBM}}$ is the way in which the inputs of honest clients to $\mathcal{F}_{\mathsf{Agg}}$ are pre-processed and the output of $\mathcal{F}_{\mathsf{Agg}}$ is post-processed (for an honest output party). More specifically, in $\mathcal{F}_{\mathsf{DBM}}$, the pre-processing of honest clients is done by the ideal functionality. Whereas in Π_{DBM}, the honest clients pre-process their own inputs and send it to $\mathcal{F}_{\mathsf{Agg}}$. This results in the same effect as honest clients perform the pre-processing do not deviate from the specification of PRE. Similarly, if the output party is honest, then the post-processing of output received from $\mathcal{F}_{\mathsf{Agg}}$ is executed as per POST in Π'. Hence, Π' in the $\mathcal{F}_{\mathsf{Agg}}$-hybrid securely realizes $\mathcal{F}_{\mathsf{DBM}}$. □

Lemma M.3. *If any protocol Π in $\mathcal{G}$-hybrid is* DP-DME*-protocol and $\widetilde{\Pi}$ securely realizes $\mathcal{G}$, then the protocol $\widehat{\Pi}$ obtained from Π by replacing the calls to $\mathcal{G}$ with the protocol $\widetilde{\Pi}$ is a* DP-DME*-protocol.*

Proof. Observe that the properties (ϵ, δ)-IND-CDP and $\nu(t)$-*DME-Usefulness* are properties over the view of the adversary $\mathsf{VIEW}_{\mathcal{A}}(\cdot)$ and the output $\mathsf{out}_{P_n}(\cdot)$. Since protocol Π is a DP-DME-protocol, it holds $(\mathsf{VIEW}_{\mathcal{A}}(e'), \mathsf{out}_{P_n}(e'))$ and $(\mathsf{VIEW}_{\mathcal{A}}(e), \mathsf{out}_{P_n}(e))$ are computationally indistinguishable where e' is obtained from the execution of $\widehat{\Pi}$ and e is obtained from the execution of Π. Hence, $\widehat{\Pi}$ is a DP-DME protocol. □

Input-Certified Secure Aggregation Protocol Π_{Agg}

Public parameters. Security parameter λ, field $\mathbb{F}$, server corruption threshold $\mathsf{t_s}$, predicate $P : \mathbb{F} \to \{0,1\}$. Let $\mathsf{pp} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{t_s}, P)$.

Parties. Clients $\mathcal{U} = \{\mathcal{U}_1, \ldots, \mathcal{U}_{\mathsf{n_c}}\}$, servers $\mathcal{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_{\mathsf{n_s}}\}$ and an output party $\mathcal{O}$.

Input & Output. $\{x_1, \ldots, x_{\mathsf{n_s}}\}$ where $x_i \in \mathbb{F}$ is client $\mathcal{U}_i$'s input. The output party $\mathcal{O}$ receives $\sum_{\mathcal{U}_i \in \mathcal{U}} P(x_i) \cdot x_i$.

Setup. Each client has a point-to-point private, authenticated channel with every server. Also, the servers have point-to-point, private authenticated channels with all other servers and a broadcast channel.

Input Sharing. $[\mathcal{U} \to \mathcal{S}]$ The sharing is as follows.

1. Each client $\mathcal{U}_i$ with input $x_i \in \mathbb{F}$ samples a set of d polynomials $f(\cdot)$ of degree $\mathsf{t_s}$ such that $f(0) = x_i$.
2. Each client $\mathcal{U}_i$ invokes the share phase of $\mathcal{F}_{\mathsf{VRS}}$ as a prover by sending the tuple $(\mathsf{Input}, \mathsf{sid}, \mathcal{U}_i, f)$ to $\mathcal{F}_{\mathsf{VRS}}$.
3. The servers participate in the above invocations of $\mathcal{F}_{\mathsf{VRS}}$ as verifiers. Each server $\mathcal{S}_j$ receives $\left(\mathsf{Output}, \mathsf{sid}, \mathcal{S}_j, \mathsf{Share}_j^{(i)}, \mathsf{happy}_j^{(i)}\right)$ from $\mathcal{F}_{\mathsf{VRS}}$ invoked by client $\mathcal{U}_i$ and outputs $\left(\mathsf{Share}_j^{(i)}, \mathsf{happy}_j^{(i)}\right)$, for all $i \in [\mathsf{n_c}]$. As per the properties of the $\mathcal{F}_{\mathsf{VRS}}$, all servers output the same happy bit i.e., $\mathsf{happy}_j^{(i)} = \mathsf{happy}_{j'}^{(i)}$ for $j, j' \in [\mathsf{n}_s]$. So, we drop the subscript j while referring to the happy bit.

Output Reconstruction. $[\mathcal{S} \to \mathcal{O}]$

1. At the end of all the invocations to $\mathcal{F}_{\mathsf{VRS}}$, the servers define a set Valid to comprise of all clients $\mathcal{U}_i$ such that $\mathsf{happy}^{(i)} = \mathsf{accept}$.
2. Each server $\mathcal{S}_j$ sums the shares it received from all clients in the set Valid i.e., $\mathsf{osh}_j = \sum_{\mathcal{U}_i \in \mathsf{Valid}} \mathsf{Share}_j^{(i)}$ and sends its output share osh_j to the output party.
3. The output party collects shares of the output osh_j from each server $\mathcal{S}_j \in \mathcal{S}$. If no share is received from the server $\mathcal{S}_j$, then the output party sets $\mathsf{osh}_j := \perp$. Finally, the output party error-corrects the vector $(\mathsf{osh}_1, \ldots, \mathsf{osh}_{n_s})$ to reconstruct Y_{agg} and sets Y_{agg} as the output.

Figure 10: An Input-Certified Secure Aggregation Protocol from VRS

Functionality $\mathcal{F}_{\mathsf{DBM}}$

The functionality $\mathcal{F}_{\mathsf{Agg}}$ communicates with the set of clients $\mathcal{U} = \{u_1, \ldots, u_{\mathsf{n_c}}\}$, an output party $\mathcal{O}$ and an adversary $\mathcal{A}$. It is parameterized by $\mathsf{n_c}, d \in \mathbb{N}$ and $P' : \mathbb{F}_q^d \to \{0,1\}$, where P' is the predicate used for certify the inputs, $\mathsf{n_c}$ is the number of clients and d denotes the size of client's input vector. The ideal functionality is as follows:

1. The inputs are processed as follows.
   - Upon receiving input $(\mathsf{Input}, \mathsf{sid}, u_i, X_i)$ from a new honest client $u_i \in \mathcal{U}$ where $X_i \in \mathbb{F}^d$, pre-process u_i's inputs as $Y_i = \mathsf{PRE}(X_i)$ and store Y_i.
   - Upon receiving input $(\mathsf{Input}, \mathsf{sid}, u_i, Y_i)$ from the adversary on behalf of some new malicious client $u_i \in \mathcal{U}$ where $X_i \in \mathbb{F}^d$, then store Y_i.
2. Compute the aggregate $\mathsf{Y}_{agg} = \sum_{u_i \in \mathcal{U}} P'(Y_i) \cdot Y_i$ (note that this is equivalent to aggregating only inputs of clients that satisfy the predicate P').
3. Upon receiving $(\mathsf{Output}, \mathsf{sid}, \mathcal{O})$ from the output party $\mathcal{O}$, send $(\mathsf{Output}, \mathsf{sid}, \mathsf{POST}(\mathsf{Y}_{agg}))$ to the output party $\mathcal{O}$ and halt.

Figure 11: Ideal Functionality for Differentially-Private Distributed Mean Estimation