# Zero-Knowledge Proofs on Secret-Shared Data via Fully Linear PCPs

Dan Boneh[1], Elette Boyle[2], Henry Corrigan-Gibbs[1], Niv Gilboa[3], and Yuval Ishai[4]

[1] Stanford University, `{dabo,henrycg}@cs.stanford.edu`
[2] IDC Herzliya, Israel, `eboyle@alum.mit.edu`
[3] Ben-Gurion University, Israel, `gilboan@bgu.ac.il`
[4] Technion, Israel, `yuvali@cs.technion.ac.il`

**Abstract.** We introduce and study the notion of *fully linear* probabilistically checkable proof systems. In such a proof system, the verifier can make a small number of linear queries that apply *jointly* to the input and a proof vector.

Our new type of proof system is motivated by applications in which the input statement is not fully available to any single verifier, but can still be efficiently accessed via linear queries. This situation arises in scenarios where the input is partitioned or secret-shared between two or more parties, or alternatively is encoded using an additively homomorphic encryption or commitment scheme. This setting appears in the context of secure messaging platforms, verifiable outsourced computation, PIR writing, private computation of aggregate statistics, and secure multiparty computation (MPC). In all these applications, there is a need for fully linear proof systems with *short proofs*.

While several efficient constructions of fully linear proof systems are implicit in the interactive proofs literature, many questions about their complexity are open. We present several new constructions of fully linear *zero-knowledge* proof systems with *sublinear* proof size for “simple” or “structured” languages. For example, in the *non-interactive* setting of fully linear PCPs, we show how to prove that an input vector $x \in \mathbb{F}^n$, for a finite field $\mathbb{F}$, satisfies a single degree-2 equation with a proof of size $O(\sqrt{n})$ and $O(\sqrt{n})$ linear queries, which we show to be optimal. More generally, for languages that can be recognized by systems of constant-degree equations, we can reduce the proof size to $O(\log n)$ at the cost of $O(\log n)$ rounds of interaction.

We use our new proof systems to construct new short zero-knowledge proofs on distributed and secret-shared data. These proofs can be used to improve the performance of the example systems mentioned above.

Finally, we observe that zero-knowledge proofs on distributed data provide a general-purpose tool for protecting MPC protocols against malicious parties. Applying our short fully linear PCPs to “natural” MPC protocols in the honest-majority setting, we can achieve unconditional protection against malicious parties with sublinear additive communication cost. We use this to improve the communication complexity of recent honest-majority MPC protocols. For instance, using any pseudorandom generator, we obtain a 3-party protocol for Boolean circuits in which the amortized communication cost is only *one bit* per AND gate per party (compared to 10 bits in the best previous protocol), matching the best known protocols for semi-honest parties.

## 1 Introduction

In this work, we develop new techniques for proving in zero knowledge statements that are *distributed* (i.e., partitioned or secret-shared) across two or more verifiers. Recall that in a standard interactive proof system [8, 10, 14, 55] a verifier holds an input $x \in \{0,1\}^*$ and a prover tries to convince the verifier that $x$ is a member of some language $\mathcal{L} \subseteq \{0,1\}^*$. We consider instead the setting in which there are *multiple* verifiers, and each verifier holds only a piece of the input, such as a share of $x$ generated using a linear secret-sharing scheme. Critically, no single verifier holds the entire input $x$. The prover, who holds the entire input $x$, must convince the verifiers, who only hold pieces of $x$, that $x \in \mathcal{L}$. At the same time, we require that the proof system be *strongly zero knowledge*: every proper subset of the verifiers should learn nothing about $x$, apart from the fact that $x \in \mathcal{L}$.

Special cases of this type of proof system appear in existing systems for anonymous messaging [40], verifiable function secret sharing [32], and systems for the private computation of aggregate statistics [39]. We observe that such proof systems also provide a powerful tool for protecting protocols for *secure multiparty computation* over *point-to-point channels* against malicious parties, analogous to the role that standard zero-knowledge proofs play in the GMW compiler [52]. Indeed, in protocols that involve point-to-point communication, the task of proving compliance with the protocol exactly requires executing a zero-knowledge proof on distributed data.

We introduce the central new abstraction of a *fully linear proof system*. Such proof systems apply not only to efficiently proving (in zero-knowledge) statements on distributed or secret-shared data, but also to data that is encrypted or committed using a linearly homomorphic system. While several efficient constructions of fully linear proof systems are implicit in the literature on interactive and probabilistically checkable proofs (in particular, the linear PCPs from [6, 51] and the interactive proofs from [54, 78] can be cast as such proof systems), many questions about their complexity are open. We present several new constructions of fully linear zero-knowledge proof systems that achieve *sublinear* proof size for "simple" or "structured" languages. Finally, we present several applications of such proof systems in the context of the motivating applications discussed above.

We now give a more detailed overview of our contributions.

**Contribution I: Fully linear proof systems.** We begin by introducing the notion of a *fully linear proof system*, which captures the information-theoretic object at the core of all of our constructions. We consider the non-interactive variant of such proof systems, called *fully linear PCPs*, and then we describe a natural extension to the interactive setting.

A fully linear PCP is a refinement of linear PCPs [6, 25, 60]. Recall that in a standard linear PCP over a finite field $\mathbb{F}$, a polynomial-time verifier holds an input $x \in \mathbb{F}^n$ and a prover produces a proof $\pi \in \mathbb{F}^m$ to the assertion that $x \in \mathcal{L}$, for some language $\mathcal{L} \subseteq \mathbb{F}^n$. The verifier checks the proof by reading $x$ and making *linear queries* (i.e., inner-product queries) to the proof $\pi$. In particular, the verifier can make a bounded number of queries to the proof of the form $q_j \in \mathbb{F}^m$, and receives answers $a_j = \langle q_j, \pi \rangle \in \mathbb{F}$.

In a fully linear PCP, we further restrict the verifier: the verifier cannot read the entire input $x$ directly, but only has access to it via linear queries. Concretely, the verifier in a fully linear PCP makes linear queries $q_j$ to the concatenated input-proof vector $(x \| \pi) \in \mathbb{F}^{n+m}$ and must accept or reject the assertion that $x \in \mathcal{L}$ based on the answers $a_j$ to these linear queries. Motivated by the applications we consider, we would also like fully linear PCPs to satisfy the following *strong zero-knowledge* requirement: the queries $q_j$ together with the answers $a_j$ reveal no additional information about $x$ other than the fact that $x \in \mathcal{L}$. This is stronger than the standard notion of zero-knowledge proofs in which $x$ is essentially public and the interaction need not hide $x$. See Section 3 for formal definitions of fully linear PCPs and their strong zero knowledge variant.

The full linearity restriction is naturally motivated by applications in which the input statement is not fully available to any single verifier, but can still be efficiently accessed via linear queries. This situation arises in scenarios where the input $x$ is distributed or secret-shared between two or more parties, or alternatively is encoded using an additively homomorphic encryption or commitment scheme. In these scenarios, verifiers can readily compute answers to public linear queries via *local* computations on their view of $x$. While fully linear PCPs can be meaningfully applied in all of the above scenarios, we will primarily focus on their applications to proofs on distributed or secret-shared data.

We stress again that in a fully linear PCP, the verifier only has *linear query access* to $x$. An interesting consequence is that even if $\mathcal{L}$ is an easy language that can be decided in polynomial time, a verifier making a bounded (e.g., constant) number of such queries typically cannot decide whether $x \in \mathcal{L}$ without the aid of a proof, even if the verifier can run in unbounded time. This makes the existence of fully linear proof systems with good parameters meaningful even for finite languages and even if, say, $\mathsf{P} = \mathsf{PSPACE}$.[1] The same fact makes possible a connection between fully linear PCPs and communication complexity [3, 66, 68]. Using this connection, we prove unconditional lower bounds on the efficiency properties of fully linear PCPs.

Different kinds of linear PCPs were used, either explicitly or implicitly, in the vast literature on succinct arguments for $\mathsf{NP}$ (see [24, 25, 27, 31, 51, 57, 60, 70, 77, 79, 80, 85, 88] and references therein). These linear PCPs, including the "Hadamard PCP" [6, 60] and ones obtained from quadratic span programs or quadratic arithmetic programs [25, 51, 76], can be cast into the fully linear framework. This fact was implicitly used in previous proof systems on committed or secret-shared

[1] This is akin to *proofs of proximity* [22], which place a more stringent restriction on the verifier's access to the input. However, unlike proofs of proximity, in fully linear PCPs the verifier is guaranteed that the input is actually in the language rather than being "close" to some input the language. Another related notion is that of a *holographic proof* [9, 59], where the verifier gets oracle access to an *encoding* of the input using an arbitrary error-correcting code.

data [11,39,41]. Our notion of fully linear PCPs provides a convenient abstraction of the properties on which such systems can be based.

**Contribution II: Shorter proofs for structured and simple languages.** When using fully linear PCPs to build zero-knowledge proof systems on distributed or secret-shared data, as discussed in Contribution IV below, the *proof length* determines the number of bits that the prover must send to the verifiers. As such, we aim to design short proofs. This goal is especially important when many different assertions are proved about the same input statement $x$. In such a scenario, the initial setup cost of distributing $x$ is amortized away. Having short fully linear PCPs yields similar efficiency benefits in the settings of encryption and commitments.

These applications motivate the need for fully linear PCPs with *short proofs*. For general $\mathsf{NP}$ relations, all known linear PCPs have size at least *linear* in the size of an arithmetic circuit recognizing the relation. In Section 4, we achieve significant length savings by designing new *sublinear* sized fully linear PCPs for languages recognized by deterministic circuits with repeated sub-structures (Theorem 11) or by a degree-2 polynomial (Corollary 13). In the latter case, we can even prove that the $O(\sqrt{n})$ complexity of our construction is optimal up to low-order terms (see full version). These and other proof systems constructed in this work satisfy the notion of strong zero knowledge discussed above.

**Theorem 1 (Informal - short fully linear PCP for a degree-2 polynomial).** *If membership in $\mathcal{L} \subseteq \mathbb{F}^n$ can be recognized by a single degree-2 polynomial, then $\mathcal{L}$ admits a fully linear PCP with strong zero knowledge that has proof length and query complexity $\tilde{O}(\sqrt{n})$ and soundness error $O(\sqrt{n}/|\mathbb{F}|)$. Furthermore, there exists a language $\mathcal{L}$ as above such that the sum of the proof length and query complexity must be $\Omega(\sqrt{n})$, even when we allow constant soundness error and do not require zero knowledge.*

See Corollary 13 for a more precise and general statement.

**Contribution III: Reducing proof size by interaction.** To further drive down the proof length, we consider a generalization of fully linear PCPs that allows multiple rounds of interaction between the prover and verifier. These *fully linear interactive oracle proofs*, or fully linear IOPs, are the linear analogue of interactive oracle proofs (IOP) [21], also known as probabilistically checkable interactive proofs [78]. We note that without the zero-knowledge requirement, several existing interactive proof systems from the literature, including the GKR protocol [53], the CMT protocol [37], and the RRR protocol [78] can be viewed as fully linear IOPs.

For the case of "well-structured" languages, we show in Section 5 that interaction can dramatically shrink the proof size, while maintaining the required strong zero-knowledge property. In particular, any language whose membership can be verified by a system of constant-degree equations over a finite field admits a fully linear IOP with strong zero-knowledge in $O(\log n)$ rounds and only $O(\log n)$ proof length, provided that the underlying field is sufficiently large. Even for degree-2 languages, this provably gives an exponential reduction in proof size over the non-interactive case.

**Theorem 2 (Informal - fully linear zero-knowledge IOPs for low-degree languages).** *Suppose $\mathcal{L} \subseteq \mathbb{F}^n$ can be recognized by a system of constant-degree equations. Then, $\mathcal{L}$ admits a fully linear IOP with strong zero knowledge, $O(\log n)$ rounds, and proof length and query complexity $O(\log n)$.*

See Theorem 15 for a more precise and general statement, including an extension to rings.

**Contribution IV: Zero-knowledge proofs on distributed or secret-shared data.** The primary motivation for our new types of proof systems is the fact that in many cases, data can be efficiently accessed via linear queries. This includes several different scenarios, but our main focus in this work is on the case of *distributed* or *secret-shared* data. (See full version for application to proofs on *encrypted* or *committed* data.) More precisely, the prover knows $x$ in its entirety and each of $k$ verifiers $V_1, \ldots, V_k$ only has a piece (or a secret share) of $x$.

In the full version we show that any fully linear PCP and IOP can be compiled into a zero-knowledge proof system on distributed or secret-shared data in the following natural way. Instead of sending a proof vector $\pi$ to a single verifier, the prover $P$ secret-shares the proof vector $\pi$ between the $k$ verifiers using a linear secret-sharing scheme. The verifiers can now locally apply each linear query to the concatenation of their share of the input $x$ and their share of $\pi$, and exchange the resulting answer shares with the other verifiers. The verifiers then reconstruct the answers to the linear queries and apply the decision predicate to decide to accept or reject $x$. We present different variants of this compiler that further optimize this approach and that achieve zero-knowledge even when up to $k - 1$ verifiers are malicious.

**Theorem 3 (Informal - distributed zero-knowledge proofs for low-degree languages on secret-shared data: malicious prover *or* verifiers).** *Suppose $\mathcal{L} \subseteq \mathbb{F}^n$ can be recognized by a system of constant-degree equations. Then, assuming ideal coin-tossing, there is an $O(\log n)$-round distributed zero-knowledge protocol for proving that $x \in \mathcal{L}$, where $x$ is additively shared between $k$ verifiers, with communication complexity $O(k \log n)$. The protocol is sound against a malicious prover and is strongly zero-knowledge against $t = k - 1$ malicious verifiers.*

We also give a Fiat-Shamir-style compiler that uses a random oracle to collapse multiple rounds of interaction into a single message sent by $P$ to each $V_j$ over a private channel, followed by a single message by each $V_j$.

Given a robust encoding (or robust secret sharing) of the input $x$, we present distributed zero-knowledge protocols that maintain their soundness even when a malicious prover colludes with $t < k/2$ malicious verifiers. In contrast, we note that previous sublinear proof systems on secret-shared data either do not attempt to protect against malicious verifiers [32], or assume a majority of honest verifiers [40]. Neither considers soundness against a malicious prover colluding with malicious verifiers.

Table 1 summarizes the communication and round complexity of the proof systems on secret-shared data for languages that frequently come up in practice, for example in the Prio system [39] for privately aggregating data, and in the Riposte [40] system for anonymous communication. The table illustrates the strong benefits of interactive fully linear proof systems over non-interactive ones.

We note that interactive proofs with distributed verifiers were recently studied in [67, 73] for the purpose of proving properties of a communication graph connecting a large number of verifiers. The relevance of the interactive proofs of GKR [53] and RRR [78] to this setting has been observed in [73]. Our focus here is quite different; we are motivated by the goal of proving in zero knowledge simple properties of data distributed among a small set of verifiers. As a result, our abstractions, constructions, and applications are very different from those in prior work [67, 73].

| Language | Proof system | Comm. complexity | Rounds |
|---|---|---|---|
| Hamming weight 1: | Prio [39] | $O(n)$ | 1 |
| $\bar{x} \in \mathbb{F}^n, \mathrm{weight}(\bar{x}) = 1$ | Theorem 15 | $O(\sqrt{n})$ | 2 |
| | Theorem 15 | $O(\log n)$ | $O(\log n)$ |
| | Implicit in [43] | $O(1)$ | 2 |
| | Riposte** [40] | $O(\sqrt{n})$ | 1 |
| | Verifiable FSS** [32] | $O(1)$ | 1 |
| $\bar{x} \in \{0, \dots, B\}^n \subseteq \mathbb{F}^n$ | Prio [39] | $O(B \cdot n)$ | 1 |
| | Theorem 15 | $O(B \cdot \sqrt{n})$ | 2 |
| | Theorem 15 | $O(B \cdot \log n)$ | $O(\log n)$ |
| $n$ Beaver triples: | Prio [39] | $O(n)$ | 1 |
| $\bar{x}, \bar{y}, \bar{z} \in \mathbb{F}^n$ where | Theorem 15 | $O(\sqrt{n})$ | 2 |
| $x_i \cdot y_i = z_i$ for all $i \in [n]$ | GKR [54] | $O(\log^2 n)$ | $O(\log^2 n)$ |
| | Theorem 15 | $O(\log n)$ | $O(\log n)$ |
| Arbitrary circuit $C$, $C(\bar{x}) = 1$ | Prio [39] | $O(n)$ | 1 |
| (size $n$, depth $d$, fan-in 2) | GKR [54] | $O(d \log n)$ | $O(d \log n)$ |

Table 1: Communication and round complexity for proof systems where the input data is secret shared among a number of parties. We assume the proofs are over a finite field $\mathbb{F}$ with $|\mathbb{F}| \gg n$. Prio [39] is a system for private data aggregation that uses proofs on secret shared data for data integrity. Riposte [40] is a system for anonymous communication that uses proofs on secret shared data to prevent data corruption. Verifiable function secret sharing (FSS) [32] enables secret sharing of simple functions.
** All systems in the table, except Riposte, verifiable FSS, and GKR, maintain zero knowledge when all but one of the verifiers are malicious. In contrast, 3-server Riposte tolerates only one corruption. Verifiable FSS tolerates only semi-honest verifiers and GKR does not provide zero-knowledge.

**Contribution V: Applications to honest-majority MPC.** We next demonstrate applications of our zero-knowledge fully linear proof systems for protecting protocols for secure multiparty computation (MPC) in the honest-majority setting against *malicious* parties, with vanishing amortized communication overhead, and without resorting to the heavy machinery of succinct (two-party) zero-knowledge argument systems for NP.

Compiling "natural" honest-majority protocols. Dating back to the work of Goldreich, Micali, and Wigderson (GMW) [52], the standard approach to secure protocol design begins by attaining semi-honest (passive) security, then compiling the protocol in some way to enforce semi-honest behavior. The GMW compiler relies on standard zero-knowledge proofs, which apply to *public* statements. As a result, it does not apply directly to the case of protocols that employ communication over secure point-to-point channels. To get around this limitation, we employ our distributed zero-knowledge proofs in the following way.

As observed in recent works, the vast majority of semi-honest MPC protocols from the literature share the following natural form:

– Up to the final exchange of messages, the protocol reveals *no* information about parties' inputs, even if parties act maliciously.
– The messages sent by a party $P_i$ in each round are *degree-2* functions (or, more generally, low-degree functions) of messages received in previous rounds.

The first property means that parties can safely execute all but the final round of the underlying protocol unchanged, and then simultaneously verify that in all prior rounds the parties acted semi-honestly. The second property means that this verification can be expressed as satisfaction of a collection of several degree-2 constraints on parties' incoming and outgoing messages. More concretely, each party $P_i$ must convince the remaining parties in zero knowledge that the statement $M_i$ consisting of all his round-by-round incoming and outgoing messages—and which is *distributed* across the remaining parties—is indeed contained within some appropriate language $\mathcal{L}_i$ verifiable by a degree-2 circuit. This is precisely the setting of our zero knowledge proofs on distributed data.

We demonstrate an approach for compiling semi-honest protocols of the above "natural" form (formally defined in the full version) in the honest-majority setting, to *malicious* security with abort, with *sublinear additive communication overhead*. This is achieved by adding a phase in the penultimate round of the base protocol, in which each party $P_i$ executes a single interactive proof on distributed data that the *entire* interaction thus far has been performed honestly. The necessary zero-knowledge protocols that we develop induce communication that is sublinear in the circuit size.

Note that while many efficient MPC protocols from the literature implement batch-verification of shared secrets by revealing random linear combinations, this technique only applies to checking *linear* relations between the secrets. Fully linear proof systems provide a powerful extension of this approach to batch-verification of *non-linear* relations with sublinear communication cost.

THE CASE OF 3-PARTY COMPUTATION. A specific motivated setting is that of 3-party computation with 1 malicious corruption (and security with abort). The task of minimizing communication in such protocols has attracted a significant amount of research effort (e.g., [4, 5, 36, 46, 50, 56, 69, 72, 74]). To date, the best protocols communicate: 2 field elements per multiplication gate per party over large fields (size comparable to $2^\sigma$ for statistical security parameter $\sigma$) [36, 74], or alternatively 10 bits per multiplication gate per party over Boolean circuits [50].

Applying our compiler to a 3-party semi-honest protocol of Katz et al. [64][2] (see also [5, 42, 45]), we obtain a 3-party protocol guaranteeing security with abort against 1 malicious party, with *1 ring element* communicated per party per multiplication (amortized over large circuit size). Our result holds over any finite field or modular arithmetic ring $\mathbb{Z}_w$; in particular, also for Boolean circuits.

**Theorem 4 (Informal - Malicious 3PC, 1 ring element/gate/party).** *There exists a 3-party protocol for securely computing any $R$-arithmetic circuit $C$ (for $R$ field of arbitrary size or $R = \mathbb{Z}_w$) with the following features:*

- *The protocol makes black-box use of any pseudorandom generator. If $R$ is a field, it also makes a black-box use of $R$.*
- *The protocol is computationally secure with abort against one malicious party.*
- *The communication complexity is $|C| + o(|C|)$ elements of $R$ per party, where $|C|$ denotes the number of multiplication and input gates in $C$.*

We also describe an application of a variant of our compiler in the more general honest majority case where $t < n/2$ for constant $n$, building from a semi-honest protocol à la Damgård and Nielsen [44]. Overall, our resulting protocol achieves malicious security with $3t/(2t + 1)$ (always $\leq 1.5$) ring elements communicated per gate per party.

## 2 A Taxonomy of Information-Theoretic Proof Systems

One of the contributions of this work is to introduce and formalize the notions of *fully linear* PCPs and IOPs. To situate these new types of proof systems in the context of prior work, we briefly survey the landscape of existing systems. This discussion will be relatively informal; see Section 3 for formal definitions of linear and fully linear proof systems.

A tremendously successful paradigm for the construction of cryptographic proof systems is the following: First, construct a proof system that provides security guarantees (soundness and possibly zero-knowledge) against *computationally unbounded* parties. We will refer to this as an "information-theoretic proof system," or sometimes as a probabilistically checkable proof (PCP). This information-theoretic system is often useless as a standlone object, since it typically makes

[2] Our compiler can analogously apply to the 3-party semi-honest protocol of Araki et al. [5]. We build on the protocol from [64] since its dealer-party structure offers a slightly simpler description within our framework and the advantage of lower online (input-dependent) cost.

idealized assumptions (such as independence between two messages or restricted access to the proof) that are difficult to enforce. Next, use cryptographic assumptions and/or an augmented model of computation (e.g., the random-oracle model [12]) to "compile" the information-theoretic proof system into one that can be directly implemented. This compiler may also provide extra desirable properties, such eliminating interaction, improved communication complexity, or sometimes even an extra zero knowledge property, at the possible cost of security against *computationally bounded* prover and/or verifier. We refer to this type of compiler as a "cryptographic compiler."

Different kinds of information-theoretic proof systems call for different cryptographic compilers. The main advantage of this separation is modularity: information-theoretic proof systems can be designed, analyzed and optimized independently of the cryptographic compilers, and their security properties (soundness and zero-knowledge) do not depend on any cryptographic assumptions. It may be beneficial to apply different cryptographic compilers to the same information-theoretic proof system, as different compilers may have incomparable efficiency and security features. For instance, they may trade succinctness for better computational complexity or post-quantum security or, more relevant to this work, apply to different representations of the input statement.

To give just a few examples of this methodology: Micali [71] uses a random oracle to compile any classical PCP into a *succinct* non-interactive argument system for $\mathsf{NP}$. As another example, Ben-Or et al. [13] compile any interactive proof system into a *zero-knowledge* interactive proof system using cryptographic commitments. Finally, Bitansky et al. [25] compile a linear PCP into a succinct non-interactive argument of knowledge (SNARK) using either a "linear-only encryption" for the designated-verifier setting or a "linear-only one-way encoding," instantiated via bilinear groups, for the public verification setting.[3] In this work we compile *fully linear* PCPs and IOPs into proofs on distributed, secret-shared, encrypted, or committed data.

## 2.1 Comparison with Other Proof Systems

In the following we survey some information-theoretic proof systems used in prior work. For simplicity, we ignore the zero-knowledge feature that is typically added to all proof systems.

Let $\mathcal{L} \subseteq \{0,1\}^*$ be a language. Speaking informally, a proof system for $\mathcal{L}$ is a pair of (possibly interactive) algorithms $(P, V)$. Both the prover $P$ and verifier $V$ take a string $x \in \{0,1\}^*$ as input (e.g., a SAT formula), and the prover's task is to convince the verifier that $x \in \mathcal{L}$ (e.g., that $x$ is satisfiable). We sometimes view $x$ as a vector over a finite field $\mathbb{F}$. We require the standard notions of *completeness* and *soundness.*

[3] For instantiating the publicly verifiable variant with bilinear groups, the linear PCP needs to have a verification predicate of algebraic degree 2. Such linear PCPs can be obtained either directly or via quadratic span programs or quadratic arithmetic programs [25, 51, 76].

In the simplest such proof system, the prover sends the verifier a single proof string $\pi$ of size $\mathsf{poly}(|x|)$, the verifier reads $x$ and $\pi$, and accepts or rejects. When the verifier is randomized and efficient, this setting corresponds to a Merlin-Arthur proof system [8]. There are a number of modifications to this basic paradigm that yield interesting alternative proof systems. In particular, we can:

- *Allow interaction between the prover and verifier.* In an interactive proof, the prover and verifier exchange many messages, after which the verifier must accept or reject. Allowing interaction may increase the power of the proof system [83] and makes it possible to provide zero-knowledge [55] in the plain model. (Alternatively, a common reference string is sufficient [26].)
- *Restrict the verifier's access to the proof.* Another way to modify the basic paradigm is to restrict the means by which the verifier interacts with the proof. In particular, we can view the proof as an oracle, and only allow the verifier to make a bounded (e.g., constant) number of queries to the proof oracle.

  In the classical PCP model [7,47,48], the proof is a string $\pi \in \Sigma^m$, for some finite alphabet $\Sigma$, and the verifier can only read a small number of symbols from the proof. On input $i$, the oracle returns the $i$th bit of the proof string $\pi$. (We call these "point queries.")

  In the linear PCP model [25,60], the proof is a vector $\pi \in \mathbb{F}^m$, for some finite field $\mathbb{F}$, and the verifier can can only make a small number of "linear queries" to the proof. That is, the proof oracle takes as input a vector $q \in \mathbb{F}^m$ and returns the inner-product $\langle \pi, q \rangle \in \mathbb{F}$.
- *Restrict the verifier's access to the input.* Yet another way to modify the basic paradigm is to restrict the verifier's access to the input $x$. In particular, we can view the *input* as an oracle, and only allow the verifier to make a bounded (e.g., constant) number of queries to the input oracle. The strong motivation for this is explained later in this section. We consider two variants.

  The model in which we view the input as a string, and only allow the verifier to make a limited number of point queries to the input, corresponds to a PCP of proximity [22]. With a few point queries, it is not possible to distinguish between an input $x \in \mathcal{L}$, and an input $x$ "close to $\mathcal{L}$" (in Hamming distance). For this reason, PCPs of proximity necessarily provide only a relaxed notion of soundness: if $x$ is "far from $\mathcal{L}$," then the verifier will likely reject.

  Alternatively, we can view the input as a vector $x \in \mathbb{F}^n$, for some finite field $\mathbb{F}$, and only allow the verifier to make a small number of linear queries to the input $x$. That is, the input oracle takes as input a vector $q \in \mathbb{F}^n$ and returns the inner-product $\langle x, q \rangle \in \mathbb{F}$. We show that this notion, introduced and studied in this work, is sufficient to provide a standard notion of soundness (unlike the relaxed notion of soundness provided by PCPs of proximity).

We now have three attributes by which we can classify information-theoretic proof systems: interactivity (yes/no), proof query type (read all/point/linear), and input query type (read all/point/linear). Taking the Cartesian product of these attributes yields 18 different possible proof systems, and we list ten of particular interest in Table 2.

| | Proof type | Queries to input | Queries to proof | Representative compilers |
|---|---|---|---|---|
| *Non-interactive* | Classical (MA) [8] | Read all | Read all | |
| | PCP [6, 7] | Read all | Point | Kilian [65], Micali [71] |
| | Linear PCP [60] | Read all | Linear | IKO [60], Pepper [82], GGPR [51], PHGR [76, 77], BCIOP [25] |
| | PCP of proximity [23] | Point | Point | Kalai & Rothblum [62] |
| | **Fully linear PCP** | Linear | Linear | *This paper* |
| *Interactive* | Interactive proof (IP) [55] | Read all | Read all | Ben Or et al. [13] |
| | IOP [21] | Read all | Point | BCS [21] |
| | Linear IOP | Read all | Linear | |
| | IOP of proximity [16, 17] | Point | Point | |
| | **Fully linear IOP** | Linear | Linear | *This paper*, Hyrax [85], vSQL [87, 88] |

Table 2: A comparison of information-theoretic proof systems. The **bolded** proof system models are ones that we introduce explicitly in this work. "Read all" refers to reading the entire data field, "Point" refers to reading a small number of cells of the data, and "Linear" refers to a small number of linear queries to the data.

For example, interactive oracle proofs (IOPs) are interactive proofs in which the verifier has unrestricted access to the input but may make only point queries to proof strings [21]. Ben-Sasson et al. [21] show how to compile such proofs into SNARGs in the random-oracle model and recent hash-based SNARGs, including Ligero [2], STARK [15], and Aurora [20] are built using this technique.

**Why fully linear proof systems?** It is often the case that the verifier only has access to an additively homomorphic *encoding* of a statement $x$, and the prover convinces the verifier that the encoded statement is true. For example the verifier may be given an additively homomorphic commitment or encryption of the statement $x$. Or the verifier may be implemented as a set of two or more servers who have a linear secret sharing of the statement $x$, or who hold different parts of $x$.

In all these settings, the verifiers can easily compute an *encoding* of the inner product of the statement $x$ with a known query vector $q$. In some cases (such as the case of encrypted or committed data), the verifiers may need the prover's help help to "open" the resulting inner products.

When we compile fully linear PCPs into proof systems on shared, encrypted, or committed data, our compilers have the same structure: the prover sends an additively homomorphic encoding of the proof to the verifier. The verifier makes linear queries to the proof and input, and (if necessary) the prover provides "openings" of these linear queries to the verifier. The verifier checks that the openings are consistent with the encodings it was given, and then runs the fully linear PCP verifier to decide whether to accept or reject the proof.

**The need for new constructions.** In current applications of PCPs and linear PCPs, the length of the proof is not a complexity metric of much relevance. For example, in the BCIOP compiler [25] for compiling a linear PCP into a succinct non-interactive argument of knowledge (SNARK), the size of the proof corresponds to the prover's running time.

If the language $\mathcal{L}$ in question is decided by circuits of size $|C|$, then having proofs of size $|C|$ is acceptable, since the prover must run in time $\Omega(|C|)$ no matter what. A similar property holds for Micali's CS proofs [71], Kilian's PCP compiler [65], the BCS compiler [21] of interactive oracle proofs, and so on.

In our compilers, the prover must materialize the entire fully linear PCP proof, encode it, and send it to the verifier. For us, the size of the fully linear PCP proof not only dictates the running time of the prover, but also dictates the number of bits that the prover must communicate to the verifier. For this reason, in our setting, minimizing the proof size is an important goal.

Furthermore, when compiling linear PCPs into SNARKs using the existing compilers [25, 58, 77] it is critical that the linear PCP verifier be expressible as an arithmetic circuit of degree two. This is because the linear PCP verification checks are essentially run "in the exponent" of a bilinear group. In contrast, the settings we consider allow for more flexibility: the arithmetic degree of the verifier typically does not play a role in the final applications, except perhaps for a possible influence on proof verification time.

**Relating fully linear PCPs to streaming proof systems.** The setting of *stream annotations* [35], introduced by Chakrabarti, Cormode, McGregor, and Thaler, restricts not only the verifier's access to the input and proof, but also the space usage of the verifier. In this model, the verifier is a space-bounded streaming algorithm: it may take a single pass over the input and proof, and must decide whether to accept or reject. For example, the verifier might be allowed only $O(\sqrt{n})$ bits of working space to decide inputs of length $n$. The *streaming interactive proof* model [38] is a generalization in which the prover and verifier may interact.

Fully linear interactive proofs naturally give rise to stream annotation proof systems. The reason is that if a fully linear PCP verifier makes $q_\pi$ linear proof queries and $q_x$ linear input queries, then the verifier can compute the responses to all of its queries by taking a single streaming pass over the input and proof while using $(q_x + q_\pi)\log_2|\mathbb{F}|$ bits of space. Thus, fully linear PCPs with small proof size and query complexity give rise to stream annotation proof systems with small proof and space requirements. Similarly, fully linear IOPs give rise to streaming interactive proofs.

The implication in the other direction does not always hold, however, since stream annotation systems do not always give rise to fully linear PCPs with good parameters. The reason is that a streaming verifier may, in general, compute some non-linear function of the input that is difficult to simulate with linear queries.

**Other proof systems.** We briefly mention a number of other important classes of proof systems in the literature that are out of scope of this discussion. *Linear* *interactive proofs* are a model of interactive proof in which each message that the prover sends is an affine function of all of the verifier's previous messages (but is not necessarily an affine function of the input) [25].

The fully linear PCP model is well matched to the problem of proving statements on data encoded with an additively homomorphic encoding, such as Paillier encryption [75] or a linear secret-sharing scheme. A different type of encoding is a *succinct* encoding, in which the prover can commit to a vector in $\mathbb{F}^m$ with a string of size sublinear in $m$ [34, 63]. Bootle et al. [30] introduce the "Ideal Linear Commitment" (ILC) model as an abstraction better suited to this setting. In the ILC proof model, the prover sends the verifier *multiple* proofs vectors $\pi_1, \dots, \pi_k \in \mathbb{F}^m$ in each round. The verifier is given a proof oracle that takes as input a vector $q \in \mathbb{F}^k$ and returns the linear combination $q^T \cdot (\pi_1 \ \dots \ \pi_k) \in \mathbb{F}^m$. It is possible to translate linear IOP proofs into ILC proofs (and vice versa) up to some looseness in the parameters. A linear IOP in which the prover sends a length-$m$ proof in each round implies an ILC proof with the same query complexity in which the prover sends $m$ proofs of length 1 in each round. An ILC proof in which the prover sends $k$ proofs of length $m$ and makes $\ell$ queries in each round implies a linear IOP with proof length $k \cdot m$ and query complexity $\ell \cdot m$. ILC-type proofs underlie the recent succinct zero-knowledge arguments of Bootle et al. [29] and Bünz et al. [33], whose security is premised on the hardness of the discrete-log problem.

Finally, another related notion from the literature is that of a *holographic proof* [9, 59], where the verifier gets oracle access to an *encoding* of the input using an arbitrary error-correcting code, typically a Reed-Muller code. Our notion of fully linear PCPs can be viewed as a variant of this model where the input is (implicitly) encoded by the Hadamard code and the proof can be accessed via *linear* queries (as opposed to point queries). In fact, our model allows a single linear query to apply *jointly* to the input and the proof.

We have not discussed multi-prover interactive proofs [14], in which multiple non-colluding provers interact with a single verifier, or more recently, multi-prover proofs in which a verifier gets access to multiple (possibly linear) proof oracles [28, 60].

**"Best-of-both-worlds" proof systems.** To conclude this section, we point to an interesting direction for future work on proof systems. A very desirable type of proof system, which is *not* listed in Table 2, would be one in which the verifier makes *linear* queries to the input and *point* queries to the proof. This type of proof system, which we call a *strongly linear* proof, achieves in some sense the "best of both worlds:" the verifier has restricted access to the input (as in a PCP of proximity or fully linear PCP) and yet achieves the standard notion of soundness (as in a classical PCP). While it is possible in principle to construct such strongly linear PCPs and IOPs by combining standard PCPs or IOPs of proximity [18, 22] with linear error-correcting codes, this generic combination may not yield the best achievable parameters.

## 3 Definitions

**Notation.** For $n \in \mathbb{N}$, let $[n] = \{1, \dots, n\}$. Let $\|$ denote concatenation, $\langle \cdot, \cdot \rangle$ inner product and $\perp$ the empty string. When $C$ is an arithmetic circuit over a finite field $\mathbb{F}$, we use $|C|$ to denote the number of multiplication gates in the circuit. When $|\mathbb{F}| > n$, we let $1, 2, \dots, n$ denote distinct nonzero field elements.

**On concrete vs. asymptotic treatment.** Since our new types of proof systems are meaningful even when all of algorithms involved are computationally unbounded, our definitions refer to languages and $\mathsf{NP}$-relations as finite objects and do not refer to running time of algorithms. All of our definitions can be naturally extended to the standard asymptotic setting of infinite languages and relations with polynomial-time verifiers, honest provers, simulators, and knowledge extractors. Our positive results satisfy these asymptotic efficiency requirements.

**Fully linear PCPs.** Our new notion of *fully* linear PCPs build upon the definitions of standard linear PCPs from Ishai et al. [60] and Bitansky et al. [25]. We start by recalling the original notion.

**Definition 5 (Linear PCP).** Let $\mathbb{F}$ be a finite field and let $\mathcal{R} \subseteq \mathbb{F}^n \times \mathbb{F}^h$ be a binary relation. A linear probabilistically checkable proof system for $\mathcal{R}$ over $\mathbb{F}$ with proof length $m$, soundness error $\epsilon$, and query complexity $\ell$ is a pair of algorithms $(P_{\mathsf{LPCP}}, V_{\mathsf{LPCP}})$ with the following properties:

- For every $(x, w) \in \mathcal{R}$, the prover $P_{\mathsf{LPCP}}(x, w)$ outputs a proof $\pi \in \mathbb{F}^m$.
- The verifier $V_{\mathsf{LPCP}}$ consists of a query algorithm $Q_{\mathsf{LPCP}}$ and a decision algorithm $D_{\mathsf{LPCP}}$. The query algorithm $Q_{\mathsf{LPCP}}$ takes no input and outputs $\ell$ queries $q_1, \dots, q_\ell \in \mathbb{F}^m$ and state information $\mathsf{st}$. The decision algorithm $D_{\mathsf{LPCP}}$ takes as input the state $\mathsf{st}$, the statement $x$, and the $\ell$ answers $\langle \pi, q_1 \rangle, \dots, \langle \pi, q_\ell \rangle \in \mathbb{F}$ to $Q_{\mathsf{LPCP}}$'s queries. It outputs "accept" or "reject."

The algorithms additionally satisfy the following requirements:

- **Completeness.** For all $(x, w) \in \mathcal{R}$, the verifier accepts a valid proof:

$$\Pr\left[D_{\mathsf{LPCP}}(\mathsf{st}, x, \langle \pi, q_1 \rangle, \dots, \langle \pi, q_\ell \rangle) = \text{“accept”} \;:\; \begin{array}{r} \pi \leftarrow P_{\mathsf{LPCP}}(x, w) \\ (\mathsf{st}, q_1, \dots, q_\ell) \leftarrow Q_{\mathsf{LPCP}}() \end{array}\right] = 1.$$

- **Soundness.** For all $x^* \notin \mathcal{L}(\mathcal{R})$, and for all false proofs $\pi^* \in \mathbb{F}^m$, the probability that the verifier accepts is at most $\epsilon$:

$$\Pr\left[D_{\mathsf{LPCP}}(\mathsf{st}, x^*, \langle \pi^*, q_1 \rangle, \dots, \langle \pi^*, q_\ell \rangle) = \text{“accept”} \;:\; (\mathsf{st}, q_1, \dots, q_\ell) \leftarrow Q_{\mathsf{LPCP}}()\right] \leq \epsilon.$$

In some applications, one also needs a **knowledge** property [25]: there exists an extractor $E_{\mathsf{LPCP}}$ such that if $V_{\mathsf{LPCP}}(x)$ accepts a proof $\pi$, then $E_{\mathsf{LPCP}}$ on input $\pi$ outputs a witness $w$ such that $(x, w) \in \mathcal{R}$. The linear PCPs we introduce in this work satisfy this property, though we only prove the simpler soundness property.

*Remark (Linear PCPs for languages).* On occasion we refer to linear PCPs for a *language* $\mathcal{L} \subseteq \mathbb{F}^n$, rather than for a binary relation $\mathcal{R} \subseteq \mathbb{F}^n \times \mathbb{F}^h$. This will typically be the case when $\mathcal{L}$ is efficiently recognizable, in which case the prover does not require an additional witness $w$. Essentially the same notions of completeness and soundness apply in this setting: if $x \in \mathcal{L}$, the verifier always accepts and for all $x \notin \mathcal{L}$ the verifier rejects except with at most $\epsilon$ probability.

We now define our main new notion of *fully linear PCPs* and their associated *strong zero knowledge* property.

**Definition 6 (Fully linear PCP - FLPCP).** We say that a linear PCP is *fully linear* if the decision predicate $D_{\mathsf{LPCP}}$ makes only linear queries to both the statement $x$ and to the proof $\pi$. More formally, the query algorithm $Q_{\mathsf{LPCP}}$ outputs queries $q_1, \ldots, q_\ell \in \mathbb{F}^{m+n}$, and state information $\mathsf{st}$. The decision algorithm $D_{\mathsf{LPCP}}$ takes as input the query answers $a_1 = \langle (x \| \pi), q_1 \rangle, \ldots, a_\ell = \langle (x \| \pi), q_\ell \rangle$, along with the state $\mathsf{st}$, and outputs an accept/reject bit.

*Remark.* If we do not restrict the running time of the linear PCP verifier and we do not restrict the manner in which the verifier can access the statement $x$, then all relations have trivial a linear PCPs: an inefficient linear PCP verifier can simply iterate over every possible witness $w$ and test whether $(x, w) \in \mathcal{R}$. To make the definition non-trivial, the standard notion of PCPs [84] (and also linear PCPs [25, 60]) restricts the verifier to run in polynomial time. In contrast, a fully linear PCP restricts the *verifier's access to the statement* $x$ by permitting the verifier to make a bounded number of linear queries to $x$. This restriction makes the definition non-trivial: even if the verifier can run in unbounded time, it cannot necessarily decide whether $x \in \mathcal{L}(\mathcal{R})$ without the help of a proof $\pi$.

**Definition 7 (Strong zero-knowledge fully linear PCPs).** A fully linear PCP is *strong honest-verifier zero knowledge* (strong HVZK) if there exists a simulator $S_{\mathsf{LPCP}}$ such that for all $(x, w) \in \mathcal{R}$, the following distributions are identical:

$$S_{\mathsf{LPCP}}() \quad \equiv \quad \left\{ \begin{matrix} (q_1, \ldots, q_\ell) \\ (\langle (x\|\pi), q_1 \rangle, \ldots, \langle (x\|\pi), q_\ell \rangle) \end{matrix} \ : \ \begin{matrix} \pi \leftarrow P_{\mathsf{LPCP}}(x, w) \\ (q_1, \ldots, q_\ell) \leftarrow Q_{\mathsf{LPCP}}() \end{matrix} \right\}.$$

*Remark.* The strong zero-knowledge property here departs from the traditional zero-knowledge notion in that it essentially requires that an honest verifier learn *nothing* about the statement $x$ by interacting with the prover, except that $x \in \mathcal{L}(\mathcal{R})$. This notion is meaningful in our applications, since the statement $x$ could be encrypted or secret-shared (for example), and thus it makes sense for a verifier to learn that $x \in \mathcal{L}(\mathcal{R})$ without learning anything else about $x$.

**Fully Linear Interactive Oracle Proofs.** In a linear PCP, the interaction between the prover and verifier is "one-shot:" the prover produces a proof $\pi$, the verifier makes queries to the proof, and the verifier either accepts or rejects the proof. We define *fully linear interactive oracle proofs* ("fully linear IOPs"), generalizing linear PCPs to several communication rounds. This sort of linear proof system is inspired by the notion of IOPs from [21, 78] (generalizing an earlier notion of interactive PCPs [61]) that use point queries instead of linear queries.

In the $i$th round of a linear IOP interaction, the prover sends the verifier a proof $\pi_i \in \mathbb{F}^m$, where $\mathbb{F}$ is a finite field and $m$ is a proof length parameter. The verifier issues linear queries to the proof $\pi_i$ and then sends a challenge $r_i \in \{0,1\}^*$ to the prover. The verifier's queries in round $i$, along with the challenges it produces, may depend on all of the messages it has seen thus far. The prover's next proof $\pi_{i+1}$ may depend on the challenge $r_i$, and all of the messages it has seen so far.

As with a linear PCP, we also introduce the notion of *fully linear IOPs*, in which the verifier makes only linear queries to the input, and define the strong zero-knowledge property in a natural way. The fully linear IOPs constructed in this work are all *public-coin* in the following sense.

**Definition 8 (Public-coin fully linear IOP).** We say that a $t$-round $\ell$-query fully linear IOP is *public coin* if it satisfies the following additional properties:

1. In every round $i \in \{1,\dots,t\}$ of interaction, first the prover provides a proof $\pi_i$ and then a *public* random challenge $r_i$ is picked uniformly at random from a finite set $\mathcal{S}_i$. (The choice of $r_i$ is made independently of the proof $\pi_i$ of the same round.) The public randomness $r_i$ can influence all proofs generated by the prover in the following rounds. Following the final round, $\ell$ queries $(q_1,\dots,q_\ell)$ (made to $x\|\pi_1\|\dots\|\pi_t$) are determined by the random challenges $(r_1,\dots,r_t)$.
2. The verifier's decision predicate is a function only of the public random challenges $(r_1,\dots,r_t)$ and the answers to the verifier's queries $(q_1,\dots,q_\ell)$.

When the first round does not involve a proof but only a random challenge $\pi_i$, we deduct $1/2$ from the number of rounds. In particular, a 1.5-round public-coin fully linear IOP is one that involves (in this order): a random challenge $r$, a proof $\pi$ (that may depend on $r$), queries $(q_1,\dots,q_\ell)$ to $x\|\pi$ that may depend on fresh public randomness $r'$, and decision based on $r, r'$ and the answers to the queries.

## 4 Constructions: Fully linear PCPs

In this section we first show how to construct fully linear PCPs from existing linear PCPs. Next, we introduce a new fully linear PCP that yields shorter proofs for languages that are recognized by arithmetic circuits with certain repeated structure; the only cost is an increase in the algebraic degree of the verifier, which is irrelevant for the main applications we consider. This new fully linear PCP is used by our fully linear IOP constructions in Section 5.

We begin by observing that the Hadamard [6, 25] and GGPR-style linear PCPs [19, 25, 51, 81], as described in the work of Bitansky et al. [25, Appendix A], satisfy our new notions of full linearity and strong zero knowledge.

| Linear PCP | Proof length | Queries | Verifier deg. | Soundness error |
|---|---|---|---|---|
| Hadamard LPCP [6,25] | $O(\|C\|^2)$ | 3 | 2 | $O(1)/\|\mathbb{F}\|$ |
| GGPR-style [51] | $O(\|C\|)$ | 4 | 2 | $O(\|C\|)/\|\mathbb{F}\|$ |
| $G$-gates (Thm. 11) | $M \cdot \deg G$ | $L+2$ | $\deg G$ | $M \cdot \deg G/(\|\mathbb{F}\| - M)$ |
| Degree-two (Cor. 13) | $O(\sqrt{\|C\|})$ | $O(\sqrt{\|C\|})$ | 2 | $O(\sqrt{\|C\|})/\|\mathbb{F}\|$ |

Table 3: A comparison of existing and new fully linear PCP constructions for satisfiability of an arithmetic circuit $C : \mathbb{F}^n \to \mathbb{F}$. Proof length measures the number of field elements in $\mathbb{F}$. For the $G$-gates construction, $G : \mathbb{F}^L \to \mathbb{F}$ is an arithmetic circuit of degree $\deg G$ and $M$ is the number of $G$-gates in the circuit $C$.

**Claim 9 (Informal).** The Hadamard linear PCP and the GGPR-based linear PCP are constant-query *fully* linear PCPs, in the sense of Definition 6. Moreover, they yield fully linear PCPs with *strong* HVZK.

We now describe a fully linear PCP for arithmetic circuit satisfiability, for circuits $C$ with a certain type of repeated structure. When applied to arithmetic circuits of size $|C|$, it can yield proofs of length $o(|C|)$ field elements. In contrast, the existing general-purpose linear PCPs in Claim 9 have proof size $\Omega(|C|)$.

This new linear PCP construction applies to circuits that contain many instances of the same subcircuit, which we call a "$G$-gate." If the arithmetic degree of the $G$-gate is small, then the resulting linear PCP is short. More formally, we define:

**Definition 10 (Arithmetic circuit with $G$-gates).** We say that a gate in an arithmetic circuit is an *affine gate* if (a) it is an addition gate, or (b) it is a multiplication gate in which one of the two input is a constant. Let $G : \mathbb{F}^L \to \mathbb{F}$ be an arithmetic circuit composed of affine gates and multiplication gates. An *arithmetic circuit with G-gates* is an arithmetic circuit composed of affine gates and $G$-gates.

The following theorem is the main result of this section. Recall that $|G|$ refers to the number of non-constant multiplication gates in the arithmetic circuit for $G$.

**Theorem 11.** *Let $C$ be an arithmetic circuit with $G$-gates over $\mathbb{F}$ such that:*
*(a) the gate $G : \mathbb{F}^L \to \mathbb{F}$ has arithmetic degree $\deg G$,*
*(b) the circuit $C$ consists of $M$ instances of a $G$-gate and any number of affine gates, and*
*(c) the field $\mathbb{F}$ is such that $|\mathbb{F}| > M \deg G$.*

*Then, there exists a fully linear PCP with strong HVZK for the relation $\mathcal{R}_C = \{(x, w) \in \mathbb{F}^n \times \mathbb{F}^h \mid C(x, w) = 0\}$ that has:*
- *proof length $h + L + M \deg G + 1$ elements of $\mathbb{F}$, where $h$ is the witness length and $L$ is the arity of the $G$-gate,*
- *query complexity $L + 2$,*
- *soundness error $M \deg G/(|\mathbb{F}| - M)$, and*

– *a verification circuit of degree* $\deg G$ *containing* $|G|$ *multiplication gates.*

*Furthermore, if we require a fully linear PCP that is not necessarily strong HVZK, then the proof length decreases to* $h + (M-1)\deg G + 1$ *elements of* $\mathbb{F}$ *and the soundness error decreases to* $M \deg G/|\mathbb{F}|$.

The proof of Theorem 11 uses the following simple fact about the linearity of polynomial interpolation and evaluation.

**Fact 12.** Let $\mathbb{F}$ be a finite field and let $\pi \in \mathbb{F}^m$. For some integer $n < |\mathbb{F}|$, let $A_1, \dots, A_n$ be affine functions that map $\mathbb{F}^m$ to $\mathbb{F}$. Define $f$ to be the polynomial of lowest-degree such that $f(i) = A_i(\pi)$ for all $i \in \{1, \dots, n\}$. Then for all $r \in \mathbb{F}$ and all choices of the $A_i$, there exists a vector $\lambda_r \in \mathbb{F}^m$ and scalar $\delta_r \in \mathbb{F}$, such that $f(r) = \langle \lambda_r, \pi \rangle + \delta_r$ for all $\pi \in \mathbb{F}^m$.

Fact 12 says that given the values of a polynomial $f$ at the points $1, \dots, n$ as affine functions of a vector $\pi \in \mathbb{F}^m$, we can express $f(r)$ as an affine function of $\pi$, and this affine function is independent of $\pi$. This follows from the fact that polynomial interpolation applied to the $n$ points $\big\{(i, A_i(\pi))\big\}_{i=1}^{n}$ followed by polynomial evaluation at the point $r$ is an affine function of $\pi$.

*Proof of Theorem 11.* The construction that proves Theorem 11 is a generalization of the linear PCP implicit in the construction used in the Prio system [39] and is closely related to a Merlin-Arthur proof system of Williams for batch verification of circuit evaluation [86]. Figure 4 gives an example of the proof construction, applied to a particular simple circuit.

Label the $G$-gates of the circuit $C$ in topological order from inputs to outputs; there are $M$ such gates in the circuit. Without loss of generality, we assume that the output of the circuit $C$ is the value on the output wire of the last $G$-gate in the circuit.

**FLPCP prover.** On input $(x, w) \in \mathbb{F}^n \times \mathbb{F}^h$, the prover evaluates the circuit $C(\cdot, \cdot)$ on the pair $(x, w)$. The prover then defines $L$ polynomials $f_1, \dots, f_L \in \mathbb{F}[X]$ such that, for every $i \in \{1, \dots, L\}$,

(i) the constant term $f_i(0)$ is a value chosen independently and uniformly at random from $\mathbb{F}$, and

(ii) for all $j \in \{1, \dots, M\}$, $f_i(j) \in \mathbb{F}$ is the value on the $i$-th input wire to the $j$-th $G$-gate when evaluating the circuit $C$ on the input-witness pair $(x, w)$.

Furthermore, the prover lets $f_1, \dots, f_L$ be the polynomials of lowest degree that satisfy these relations. Observe that each of the polynomials $f_1, \dots, f_L$ has degree at most $M$.

Next, the prover constructs a proof polynomial $p = G(f_1, \dots, f_L) \in \mathbb{F}[X]$. By construction of $p$, we know that, for $j \in \{1, \dots, M\}$, $p(j)$ is the value on the output wire from the $j$-th $G$-gate in the evaluation of $C(x, w)$. Moreover, $p(M) = C(x, w)$. Let $d$ be the degree of the polynomial $p$ and let $c_p \in \mathbb{F}^{d+1}$ be the vector of coeffcients of $p \in \mathbb{F}[X]$. By construction, the degree of $p$ satisfies $d \le M \deg G$.

The prover outputs $\pi = (w,\ f_1(0), \ldots, f_L(0),\ c_p) \in \mathbb{F}^{h+L+d+1}$ as the linear PCP proof.

(*Note:* If we do not require strong HVZK to hold, then the prover need not randomize the constant terms of the polynomials $f_1, \ldots, f_L$. In this case, the prover does not include the values $f_1(0), \ldots, f_L(0)$ in the proof, and the degree of the polynomial $p$ decreases to $(M-1)\deg G$. Thus, if we do not require strong HVZK, the proof length falls to $h + (M-1)\deg G + 1$.)

**FLPCP queries.** We can parse the (possibly maliciously crafted) proof $\pi \in \mathbb{F}^{h+L+d+1}$ as: a purported witness $w' \in \mathbb{F}^h$, the values $(z'_1, \ldots, z'_L) \in \mathbb{F}^L$ representing the constant terms of some polynomials $f'_1, \ldots, f'_L$, and the coefficients $c'_p \in \mathbb{F}^{d+1}$ of a polynomial $p' \in \mathbb{F}[X]$ of degree at most $d$. If the proof is well-formed, the polynomial $p'$ is such that $p'(j)$ encodes the output wire of the $j$th $G$-gate in the circuit $C(\cdot,\cdot)$ when evaluated on the pair $(x, w')$.

Given $p'$, we define $L$ polynomials $f'_1, \ldots, f'_L \in \mathbb{F}[X]$ such that:

(i) the constant term satisfies $f'_i(0) = z'_i$, where $z'_i$ is the value included in the proof $\pi'$, and

(ii) $f'_i(j) \in \mathbb{F}$ is the value on the $i$-th input wire to the $j$-th $G$-gate in the circuit, under the purported assignment of values to the output wires of the $G$-gates implied by the polynomial $p'$ and witness $w'$.

More precisely, we define $f'_i(j)$ inductively: The value on the $i$th input wire to the $j$th $G$-gate in the circuit $C(x, w')$ is some affine function $A_{ij}$ of

– the input $x \in \mathbb{F}^n$,
– the purported witness $w' \in \mathbb{F}^h$, and
– the purported outputs of the first $j-1$ $G$-gates in the circuit: $p'(1), \ldots, p'(j-1) \in \mathbb{F}$.

So, for all $i \in \{1, \ldots, L\}$, we define $f'_i$ to be the polynomial of least degree satisfying:

$$\begin{aligned} &f'_i(0) = z'_i \\ &f'_i(j) = A_{ij}(x, w', p'(1), ..., p'(j-1)) \qquad \text{for} \quad 1 \le j \le M, \end{aligned}$$

where $A_{ij}$ is a fixed affine function defined by the circuit $C$.

The verifier's goal is to check that:

1. $p' = G(f'_1, \ldots, f'_L)$, and,
2. the circuit output $p'(M)$ satisfies $p'(M) = 0$.

As we argue below, the first condition ensures that $p'(M)$ is equal to the output of the circuit $C(x, w')$. The second check ensures that the output is 0.

To implement the first check, the verifier samples a random point $r \xleftarrow{R} \mathbb{F} \setminus \{1, \ldots, M\}$ and outputs query vectors that allow evaluating $p'$ and $f'_1, \ldots, f'_L$ at the point $r$. (For the honest-verifier zero knowledge property to hold, it is important that we exclude the set $\{1, \ldots, M\}$ from the set of choices for $r$.) The verifier has linear access to the input $x$, witness $w'$, constant terms $z' = (z'_1, \ldots, z'_L)$, and the coefficients $c'_p \in \mathbb{F}^{d+1}$ of the polynomial $p'$. Hence, using

Fact 12, it follows that the query algorithm can compute vectors $\lambda_1, \ldots, \lambda_L \in \mathbb{F}^{n+h+L+d+1}$ and scalars $\delta_1, \ldots, \delta_L \in \mathbb{F}$ such that $f_i'(r) = \langle \lambda_i,\ (x\|w'\|z'\|c') \rangle + \delta_i$ for $i = 1, \ldots, L$, where $r \in \mathbb{F}$ is the random point chosen above. Similarly, the query algorithm can compute a vector $\lambda \in \mathbb{F}^{n+h+L+d+1}$ such that $p'(r) = \langle \lambda,\ (x\|w'\|z'\|c') \rangle$.

The verifier can execute the second check, to ensure that $p'(M) = 0$, with a single linear query.

**FLPCP decision.** The decision algorithm takes as input the state value $r \in \mathbb{F} \setminus \{1, \ldots, M\}$, along with the query answers $a, a_1, \ldots, a_L, b \in \mathbb{F}$, where $a = p'(r)$, $a_i = f_i'(r)$ for $i \in \{1, \ldots, \ell\}$, and $b = p'(M)$. The verifier accepts if $a = G(a_1, \ldots, a_L)$ and $b = 0$.

**Security arguments.** We show completeness, soundness, and strong HVZK.

*Completeness.* If the prover is honest, then $p' = G(f_1', \ldots, f_L')$ and $p'(M) = 0$ by construction. The verifier will always accept in this case.

*Soundness.* Fix a circuit $C$, a statement $x \in \mathbb{F}^n$, and a proof $\pi' \in \mathbb{F}^{h+L+d+1}$. We show that if $x \notin \mathcal{L}(\mathcal{R}_C)$ then the verifier accepts with probability at most $M \deg G/(|\mathbb{F}| - M)$.

As in the description of the query algorithm, we can view:

– the first $h$ elements of the proof as a witness $w' \in \mathbb{F}^h$,

– the next $L$ elements of the proof as constant terms $z_1', \ldots, z_L' \in \mathbb{F}$, and

– the latter elements as the coefficients of a polynomial $p'$ of degree at most $d \leq M \deg G$.

We may assume that $p'(M) = 0$, since otherwise the verifier always rejects. In the discussion that follows, let the polynomials $f_1', \ldots, f_L'$ be the ones defined in the description of the linear PCP query algorithm.

We claim that if for all $j \in \{1, \ldots, M\}$, it holds that $p'(j) = G(f_1'(j), \ldots, f_L'(j))$, then for all $j \in \{1, \ldots, M\}$, $p'(j)$ encodes the value of the output wire of the $j$th $G$-gate in the circuit $C$ when evaluated on input $(x, w')$.

We prove this claim by induction on $j$:

– *Base case ($j = 1$).* The values $(f_1'(1), \ldots, f_L'(1))$ depend only on the pair $(x, w')$. By construction, the values $(f_1'(1), \ldots, f_L'(1))$ are exactly the values of the input wires to the first $G$-gate in the evaluation of $C(x, w')$. Then if $p'(1) = G(f_1'(1), \ldots, f_L'(1))$, $p'(1)$ encodes the value on the output wire of the first $G$-gate.

– *Induction step.* Assume that, for all $k \in \{1, \ldots, j-1\}$, $p'(k) = G(f_1'(k), \ldots, f_L'(k))$. Then, by the induction hypothesis, $(p'(1), \ldots, p'(j-1))$ are the values on the output wires of the first $j-1$ $G$-gates of $C$, when evaluated on $(x, w')$.

The values $(f_1'(j), \ldots, f_L'(j))$ are affine functions of $x$, $w'$ and the values $p'(1), \ldots, p'(j-1)$. Then, by construction of the polynomials $(f_1', \ldots, f_L')$, the values $(f_1'(j), \ldots, f_L'(j))$ encode the values on the input wires to the $j$-th $G$-gate in the evaluation of the circuit $C(x, w')$. Finally, if we assume that

$p'(j) = G(f'_1(j), \ldots, f'_L(j))$, then $p'(j)$ must be the value on the output wire of the $j$th $G$-gate.

We have thus proved the induction step.

This completes the proof of the claim.

If $p'(M) = 0$ (as we have assumed), but there exists no witness $w'$ such that $C(x, w') = 0$, then $p'(M)$ does not encode the output value of the $M$th $G$-gate in the evaluation of the circuit $C(x, w')$. By the claim just proved, this implies that for some $j^* \in \{1, \ldots, M\}$, $p'(j^*) \neq G(f'_1(j^*), \ldots, f'_L(j^*))$. Thus, when we view $p', f'_1, \ldots, f'_L \in \mathbb{F}[X]$ as univariate polynomials, we have that $p' \neq G(f'_1, \ldots, f'_L)$.

Now, if $p' \neq G(f'_1, \ldots, f'_L)$ then $p' - G(f'_1, \ldots, f'_L) \in \mathbb{F}[X]$ is a non-zero univariate polynomial of degree at most $M \deg G$. Such a polynomial can have at most $M \deg G$ roots over $\mathbb{F}$. Therefore the probability, over the verifier's random choice of $r \xleftarrow{\text{R}} \mathbb{F} \setminus \{1, \ldots, M\}$, that $p'(r) - G(f'_1(r), \ldots, f'_L(r)) = 0$ is at most $M \deg G/(|\mathbb{F}| - M)$. We conclude that the verifier accepts a false proof with probability at most $M \deg G/(|\mathbb{F}| - M)$.

See full version for a proof that the construction satisfies strong HVZK. $\square$

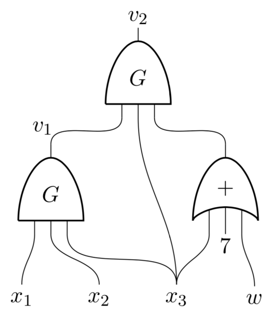
If we define the $G$-gate to be a multiplication gate, so that $\deg G = 2$, then the construction of Theorem 11 matches the complexity of the GGPR-based linear PCP [51, 81] and provides what is essentially an alternative formulation of that proof system. In contrast, if $\deg G \ll |G|$, then this construction can yield significantly shorter proofs than the GGPR-based linear PCP, at the cost of increasing the algebraic degree of the verifier from 2 to $\deg G$.

*Remark.* We can generalize Theorem 11 to handle circuits with many distinct repeated subcircuits $G_1, \ldots, G_q$ with $M_i$ instances of each gate $G_i : \mathbb{F}^{L_i} \to \mathbb{F}$, for $i \in \{1, \ldots, q\}$. The resulting fully linear PCP with strong HVZK has proof length at most $h + (\sum_{i=1}^q L_i) + (\sum_{i=1}^q M_i \deg G_i) + q$ elements of $\mathbb{F}$, query complexity $1 + \sum_{i=1}^q (L_i + 1)$, a verifier of algebraic degree $\max_i \deg G_i$, and soundness error $\sum_{i=1}^q \big(M_i \deg G_i/(|\mathbb{F}| - M_i)\big)$.

*Remark.* To get good soundness when applying the proof system of Theorem 11, the field $\mathbb{F}$ must be such that $|\mathbb{F}| \gg M \deg G$. In many applications, the input $x \in \mathbb{F}^n$ is a vector in a small field, such as the binary field $\mathbb{F}_2$. In this case, we apply Theorem 11 by lifting $x$ into an extension field $\widetilde{\mathbb{F}}$ of $\mathbb{F}$, and carrying out the linear PCP operations in the extension.

The randomization technique we use to achieve honest-verifier zero-knowledge in Theorem 11 is inspired by the one that appears in the work of Bitansky et al. [25] for achieving HVZK in the Hadamard linear PCP construction.

**Application: Short Proofs for Degree-Two Relations.** As an application of Theorem 11 we demonstrate a special-purpose fully linear PCP for relations recognized by arithmetic circuits of degree two. When applied to an arithmetic circuit $C : \mathbb{F}^n \times \mathbb{F}^h \to \mathbb{F}$, we obtain a proof that consists of only $O(h + \sqrt{n})$ field elements and whose query complexity is only $O(\sqrt{n + h})$. For general-purpose



**Circuit.** An example circuit $C(x_1, x_2, x_3, w)$ using an arbitrary three-input $G$-gate. The circuit takes as input the vector $(x_1, x_2, x_3) \in \mathbb{F}^3$, and a witness $w \in \mathbb{F}$. The circuit $C$ outputs $v_2$, the value on the output wire of the topologically last $G$-gate.

**Linear PCP proof.** Using Theorem 11, we construct a fully linear PCP proof $\pi$ that the input $(x_1, x_2, x_3) \in \mathbb{F}^3$ is in the language recognized by $C$. That is, the prover asserts that there exists a witness $w \in \mathbb{F}$ such that $C(x_1, x_2, x_3, w) = 0 \in \mathbb{F}$.

The prover first constructs three polynomials $f_1, f_2, f_3$. The value $f_i(j)$ encodes the value on the $i$-th input to the $j$-th $G$-gate, in topological order from inputs to outputs. The constant terms are random elements $z_1, z_2, z_3 \xleftarrow{R} \mathbb{F}$. That is:

| | | |
|---|---|---|
| $f_1(0)=z_1$ | $f_2(0)=z_2$ | $f_3(0)=z_3$ |
| $f_1(1)=x_1$ | $f_2(1)=x_2$ | $f_3(1)=x_3$ |
| $f_1(2)=v_1=G(x_1,x_2,x_3)$ | $f_2(2)=x_3$ | $f_3(2)=x_3+w+7$ |

Next, the prover constructs the polynomial $p$, which satisfies $p = G(f_1, f_2, f_3)$, and which has degree at most $d = 2 \deg G$. Notice that for $j \in \{1, 2\}$, $p(j)$ is the value on the output wire of the $j$-th $G$-gate. Letting $d = 3 \deg G$, we can write the values of $p$ as:

$$p(0)=G(f_1(0),f_2(0),f_3(0))=G(z_1,z_2,z_3)$$
$$p(1)=G(f_1(1),f_2(1),f_3(1))=v_1=G(x_1,x_2,x_3)$$
$$p(2)=G(f_1(2),f_2(2),f_3(2))=v_2=G(v_1,x_3,x_3+w+7)$$
$$p(3)=G(f_1(3),f_2(3),f_3(3))$$
$$\vdots$$
$$p(d)=G(f_1(d),f_2(d),f_3(d))$$

The linear PCP proof $\pi$ consists of the elements: $(w, z_1, z_2, z_3, \bar{p}) \in \mathbb{F}^{L+d+2}$, where $\bar{p} \in \mathbb{F}^{d+1}$ is the vector of coefficients of the polynomial $p$.

Fig. 4: An example of the fully linear PCP proof of Theorem 11.

linear PCPs, such as the Hadamard or GGPR-based linear PCPs, the proof length plus query complexity is much larger: $\Omega(n+h)$.

A special case of this proof yields a linear PCP for the language of vectors whose inner product is equal to a certain value. To give one application of such a proof system: Given encryptions of two sets, represented by their characteristic vectors, this proof system would allow a prover to succinctly show that the sets are disjoint.

This construction also reveals the close connection between fully linear PCPs and communication complexity. Without zero knowledge, this proof protocol boils down to the Merlin-Arthur communication complexity protocol of Aaronson and Wigderson [1]. Furthermore, as we show in the full version of this work, we can use lower bounds on the communication complexity of inner-product to show that this fully linear PCP construction has essentially optimal parameters.

**Corollary 13 (FLPCP for degree-two circuits).** *Let $\mathbb{F}$ be a finite field, let $C : \mathbb{F}^n \times \mathbb{F}^h \to \mathbb{F}$ be an arithmetic circuit of degree two, and let $\mathcal{R}_C = \{(x,w) \in \mathbb{F}^n \times \mathbb{F}^h \mid C(x,w) = 0\}$. There is a fully linear PCP with strong HVZK for $\mathcal{R}_C$ that has proof length $h + O(\sqrt{n+h})$ elements of $\mathbb{F}$, query complexity $O(\sqrt{n+h})$, a verifier of algebraic degree 2, and soundness error $\frac{O(\sqrt{n+h})}{|\mathbb{F}|-\lceil\sqrt{n+h}\rceil}$.*

The idea of Corollary 13 is that any degree-two circuit $C : \mathbb{F}^n \to \mathbb{F}$ can be expressed as a circuit that computes an inner-product of dimension-$n$ vectors, along with some number of affine gates. This property is special to degree-two circuits—the idea does not easily generalize to circuits of higher constant degree.

*Proof of Corollary 13.* Without loss of generality we can assume that $C$ implements a quadratic form $C(x,w) = (x\|w)^T \cdot A \cdot (x\|w)$ for some matrix $A \in \mathbb{F}^{(n+h)\times(n+h)}$. Indeed, a proof system for quadratic forms yields a proof system for any circuit of degree 2. We can re-write $C(x,w)$ as the inner-product of the vectors $y = (x\|w)$ and $z = A \cdot (x\|w)$ in $\mathbb{F}^{n+h}$. Hence, it suffices to design a fully linear PCP for the inner-product relation $\mathcal{R}'_C = \big\{(x,w) \in \mathbb{F}^n \times \mathbb{F}^h \mid \big\langle (x\|w)\ ,\ A \cdot (x\|w) \big\rangle = 0 \big\}$.

Let $L^2$ be the closest perfect square greater than or equal to $n+h$, and pad the vectors $y = (x\|w)$ and $z = A(x\|w)$ with zeros so that both are in $\mathbb{F}^{(L^2)}$. Next, arrange the vector $y$ into a matrix $Y \in \mathbb{F}^{L \times L}$, and arrange $z$ into a matrix $Z \in \mathbb{F}^{L\times L}$ in the same way. Then $C(x,w) = \langle y, z\rangle = \text{trace}(Y \cdot Z^T)$.

Because the trace is a linear function, we can compute $C(x,w)$ using a circuit $C'$ consisting of only addition gates and a total of $L$ gates $G : \mathbb{F}^L \times \mathbb{F}^L \to \mathbb{F}$ defined as $G(u,v) = \langle u, v\rangle$ for $u, v \in \mathbb{F}^L$. Clearly $\deg G = 2$ and $L = O(\sqrt{n+h})$. Applying Theorem 11 to this $G$-gate circuit gives a fully linear PCP for $\mathcal{R}_{C'}$ with strong HVZK with the parameters stated in the corollary, as required. The proof needs at most $2L$ additional linear queries to verify that the padding in $y$ and $z$ is all zero, but this does not change the parameters in the corollary. $\square$

*Remark.* A simple extension of Corollary 13 yields a two-round (in fact, 1.5-round) fully linear IOP for relations recognized by general degree-two circuits

$C : \mathbb{F}^n \times \mathbb{F}^h \to \mathbb{F}^k$, for $k \geq 1$. To sketch the idea behind this extension, write the circuit $C$ as $C(x) = (C_1(x), C_2(x), \ldots, C_k(x)) \in \mathbb{F}^k$, where each $C_i$ is a degree-two circuit. In the first round of the protocol, the verifier sends a random value $r \in \mathbb{F}$. Then the prover and verifier define the degree-two circuit $C_r(x) = \sum_{i=1}^k r^i \cdot C_i(x) \in \mathbb{F}$. The prover then uses the fully linear PCP of Corollary 13 to convince the verifier that $C_r$ accepts the input $x \in \mathbb{F}^n$. The efficiency parameters match those of the corollary, except that the soundness error increases by an additive term $k/|\mathbb{F}|$ to account for the failure event that some $C_i(x)$ outputs a non-zero value and yet the sum $C_r(x)$ is zero. See Theorem 15 for a more general version of this protocol.

**Application: Short Proofs for Parallel-Sum Circuits.** As a second application of Theorem 11, we give a special-purpose fully linear PCP for languages recognized by circuits that take as input a vector $x \in \mathbb{F}^n$ and:
– apply an affine transformation to the input,
– apply the same sub-circuit $C : \mathbb{F}^L \to \mathbb{F}$ in parallel to each block of $L$ values,
– sum the outputs of the $C$ circuits.

More formally, let $C : \mathbb{F}^L \to \mathbb{F}$ be an arithmetic circuit. Let $A : \mathbb{F}^n \to \mathbb{F}$ and $A_1, \ldots, A_M : \mathbb{F}^n \to \mathbb{F}^L$ be affine functions. This linear PCP construction applies to the language of values $x \in \mathbb{F}^n$ such that $\sum_{i=1}^M C(A_i(x)) = A(x)$.

**Corollary 14 (FLPCP for parallel-sum circuits).** *Let $C : \mathbb{F}^L \to \mathbb{F}$ be an arithmetic circuit over $\mathbb{F}$ that has arithmetic degree $\deg C$. Let $A : \mathbb{F}^n \to \mathbb{F}$ and $A_1, \ldots, A_M \in \mathbb{F}^n \to \mathbb{F}^L$ be affine functions. Then, there exists a strong HVZK fully linear PCP for the language $\mathcal{L}_{C,A,A_1,\ldots,A_M} = \{x \in \mathbb{F}^n \mid \sum_{i=1}^M C(A_i(x)) = A(x)\}$ that has:*
– *proof length $O(\sqrt{M} \cdot (L + \deg C))$ elements of $\mathbb{F}$,*
– *query complexity $O(\sqrt{M} \cdot L)$,*
– *soundness error $\frac{\sqrt{M} \cdot \deg C}{|\mathbb{F}| - \sqrt{M}}$, and*
– *an arithmetic verification circuit of degree $\deg C$ containing $O(\sqrt{M} \cdot |C|)$ multiplication gates.*

*Proof of Corollary 14.* We define an appropriate $G$-gate and then invoke Theorem 11. Assume that $M$ is a perfect square, since otherwise we can pad $M$ up to the nearest square. The gadget $G : \mathbb{F}^{\sqrt{M}L} \to \mathbb{F}$ applies the circuit $C$ to $\sqrt{M}$ blocks of $L$ inputs. So, on input $(\bar{x}_1, \ldots, \bar{x}_{\sqrt{M}}) \in \mathbb{F}^{\sqrt{M}L}$, where $\bar{x}_j \in \mathbb{F}^L$ for all $j \in \{1, \ldots, \sqrt{M}\}$, the $G$-gate outputs:

$$G(\bar{x}_1, \ldots, \bar{x}_{\sqrt{M}}) \stackrel{\text{def}}{=} \sum_{j=1}^{\sqrt{M}} C(\bar{x}_j) \quad \in \mathbb{F}. \tag{1}$$

Then the language $\mathcal{L}_{C,A,A_1,\ldots,A_M}$ is recognized by a circuit containing $M' = \sqrt{M}$ instances of the $G$-gate, along with some number of affine gates. Applying Theorem 11 using this $G$-gate yields a fully linear PCP as required. □

## 5 Constructions: Fully Linear Interactive Oracle Proofs

In this section, we describe an extension of our fully linear PCPs to fuller linear interactive oracle proofs (linear IOPs). These extra rounds of interaction can buy efficiency improvements in total proof length and verifier time.

For example, a corollary of our general construction gives an $O(\log n)$ round strong HVZK fully linear IOP for proving that a vector $x \in \mathbb{F}^n$ consists entirely of $0/1$ entries, where the proof size is only $O(\log n)$ field elements. In comparison, linear PCPs yield proofs of size $\Omega(n)$.

Several protocols from the literature, including notably the "Muggles" protocol of Goldwasser, Kalai, and Rothblum [53, 54] are implicitly linear IOPs. See full version for connections between our notion and these protocols.

**A Recursive Linear IOP for Parallel-Sum Circuits.** Corollary 14 gives a linear PCP for "parallel-sum" circuits whose length grows as the square root of the degree of parallelism. Here, we show that by increasing the number of rounds between the prover and verifier, we can decrease the proof size to *logarithmic* in the degree of parallelism. The key observation is that in Corollary 14, the linear PCP verifier is itself a parallel-sum circuit. So rather than having the verifier evaluate this circuit on its own, the verifier can outsource the work of evaluating the verification circuit to the prover. The prover then uses a secondary linear PCP to convince the verifier that it executed this step correctly.

To get the optimal bounds we rebalance the parameters used in the proof of Corollary 14. Instead of a $G$-gate containing $\sqrt{M}$ copies of $C$, as in (1), we use a $G$-gate containing $M/2$ copies of $C$, and then recursively verify one input/output pair for that $G$-gate.

A useful application of this technique is to the case of "low-degree languages," namely languages in which membership can be checked by a system of low-degree equations. The following theorem, whose proof appears in the full version, describes fully linear IOPs for such low-degree languages, over both finite fields and rings of the form $R = \mathbb{Z}_w$.

**Theorem 15 (ZK-FLIOP for low-degree languages).** *Let $R$ be a ring, let $C : R^n \to R^m$ be an arithmetic multi-output circuit of degree $d$ defined by $C(x) = (C_1(x), \ldots, C_m(x))$ and let $M$ be the number of distinct monomials in the representation of $C_1, \ldots, C_m$ as polynomials. Let $\mathcal{L}_C = \{x \in R^n \mid C(x) = 0^m\}$ and let $\epsilon$ be a required soundness error bound. Then, there is a fully linear IOP $\Pi$ over $R$ with strong HVZK for the language $\mathcal{L}_C$ that has the following efficiency features.*

- **Degree $d = 2$, constant rounds:** *If $d = 2$ then $\Pi$ has 1.5 rounds, proof length $O(\eta\sqrt{n})$, challenge length $O(\eta)$, and query complexity $O(\sqrt{n})$, where $\eta = \log_{|R|}((m + \sqrt{n})/\epsilon)$ if $R$ is a finite field or $\eta = \log_2((m + \sqrt{n})/\epsilon)$ if $R = \mathbb{Z}_{2^k}$. The computational complexity is $\tilde{O}(M)$*
- **Degree $d$, logarithmic rounds:** *If $d \geq 2$ then $\Pi$ has $O(\log M)$ rounds, proof length $O(\eta d \log M)$, challenge length $O(\eta)$, and query complexity $O(\log M)$,*

*where* $\eta = \log_{|R|}((m + d\log M)/\epsilon)$ *if* $R$ *is a finite field or* $\eta = \log_2((m + d\log M)/\epsilon)$ *if* $R = \mathbb{Z}_{2^k}$*. The computational complexity is* $\tilde{O}(dM)$*.*

**Trading communication for computation.** Most of our motivating applications involve low-degree verification circuits that have constant output locality. For instance, this is the case for checking that $x \in \{0,1\}^n$ or for languages corresponding to standard MPC protocols (e.g., checking Beaver triples). In this case, we can reduce computational cost while maintaining sublinear communication via the following simple tradeoff technique. Chop the $m$ outputs into blocks of size $\ell$, viewing each block as a low-degree circuit with $O(\ell)$ inputs and $\ell$ outputs, and apply a separate FLIOP to each block. This gives a smooth tradeoff between communication and computation, which may be useful for tuning concrete efficiency depending on the available bandwidth and computational power.

## 6 Conclusions

We have demonstrated that fully linear proof systems capture many existing techniques for zero-knowledge proof on secret-shared, encrypted, or committed data. We presented new constructions of zero-knowledge fully linear PCPs and IOPs for "simple" languages with sublinear proof size, and demonstrated the usefulness of such proof systems to protecting secret-sharing based MPC protocols against malicious parties with low communication overhead.

Despite some progress obtained in this work and in prior related works, there is a lot more to understand about the power of (fully) linear PCPs and their interactive variants. We mention a couple of concrete open questions:

– To what extent are the tradeoffs we obtain for low-degree languages optimal? In particular, is there a linear PCP of size $o(n)$ for the language $\mathcal{L}_{\{0,1\}^n} \overset{\mathsf{def}}{=} \{x \in \mathbb{F}^n \mid x \in \{0,1\}^n\}$? Our sublinear constructions require interaction.

– Are there linear PCPs for general arithmetic circuit satisfiability with constant query complexity and proof size sublinear in the circuit size? In the full version, we show a lower bound result that unconditionally rules out such succinct *fully* linear PCPs. *Standard* PCPs with succinctness properties cannot exist unless the polynomial hierarchy collapses [49]. Does the same restriction apply to general linear PCPs?

**Acknowledgements.** We thank Shai Halevi, Ariel Nof, Ron Rothblum, David J. Wu, and Eylon Yogev for helpful discussions and comments. Justin Thaler gave us many useful references to related work on sum-check-based proof protocols and helped us understand the relationship between those protocols and our own.

E. Boyle, N. Gilboa, and Y. Ishai supported by ERC grant 742754 (project NTSC). E. Boyle additionally supported by ISF grant 1861/16 and AFOSR Award FA9550-17-1-0069. N. Gilboa additionally supported by ISF grant 1638/15, and a grant by the BGU Cyber Center. Y. Ishai additionally supported by ISF grant 1709/14, NSF-BSF grant 2015782, and a grant from the Ministry of Science and

Technology, Israel and Department of Science and Technology, Government of India. D. Boneh and H. Corrigan-Gibbs are supported by NSF, ONR, DARPA, the Simons Foundation, CISPA, and a Google faculty fellowship.

## References

1. Aaronson, S., Wigderson, A.: Algebrization: A new barrier in complexity theory. ACM Transactions on Computation Theory (TOCT) 1(1), 2 (2009)
2. Ames, S., Hazay, C., Ishai, Y., Venkitasubramaniam, M.: Ligero: Lightweight sublinear arguments without a trusted setup. In: CCS (2017)
3. Andrew, C.C.Y.: Some complexity questions related to distributed computing. In: STOC (1979)
4. Araki, T., Barak, A., Furukawa, J., Lichter, T., Lindell, Y., Nof, A., Ohara, K., Watzman, A., Weinstein, O.: Optimized honest-majority MPC for malicious adversaries - breaking the 1 billion-gate per second barrier. In: IEEE Symposium on Security and Privacy (2017)
5. Araki, T., Furukawa, J., Lindell, Y., Nof, A., Ohara, K.: High-throughput semi-honest secure three-party computation with an honest majority. In: ACM CCS (2016)
6. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M.: Proof verification and the hardness of approximation problems. Journal of the ACM 45(3), 501–555 (1998)
7. Arora, S., Safra, S.: Probabilistic checking of proofs: A new characterization of NP. Journal of the ACM 45(1), 70–122 (1998)
8. Babai, L.: Trading group theory for randomness. In: STOC (1985)
9. Babai, L., Fortnow, L., Levin, L.A., Szegedy, M.: Checking computations in polylogarithmic time. In: STOC (1991)
10. Babai, L., Moran, S.: Arthur-merlin games: a randomized proof system, and a hierarchy of complexity classes. Journal of Computer and System Sciences 36(2), 254–276 (1988)
11. Backes, M., Barbosa, M., Fiore, D., Reischuk, R.M.: ADSNARK: nearly practical and privacy-preserving proofs on authenticated data. In: 2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015 (2015)
12. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: CCS (1993)
13. Ben-Or, M., Goldreich, O., Goldwasser, S., Håstad, J., Kilian, J., Micali, S., Rogaway, P.: Everything provable is provable in zero-knowledge. In: CRYPTO (1988)
14. Ben-Or, M., Goldwasser, S., Kilian, J., Wigderson, A.: Multi-prover interactive proofs: How to remove intractability assumptions. In: STOC (1988)
15. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046 (2018)
16. Ben-Sasson, E., Chiesa, A., Forbes, M.A., Gabizon, A., Riabzev, M., , Spooner, N.: On probabilistic checking in perfect zero knowledge. In: Electronic Colloquium on Computational Complexity (ECCC). No. 156 (2016)
17. Ben-Sasson, E., Chiesa, A., Gabizon, A., Riabzev, M., Spooner, N.: Interactive oracle proofs with constant rate and query complexity. In: ICALP (2017)
18. Ben-Sasson, E., Chiesa, A., Gabizon, A., Riabzev, M., Spooner, N.: Interactive oracle proofs with constant rate and query complexity. In: 44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland (2017)
19. Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E., Virza, M.: SNARKs for C: Verifying program executions succinctly and in zero knowledge. In: CRYPTO (2013)
20. Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for R1CS. Cryptology ePrint Archive, Report 2018/828 (2018), https://eprint.iacr.org/2018/828
21. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Theory of Cryptography Conference (2016)
22. Ben-Sasson, E., Goldreich, O., Harsha, P., Sudan, M., Vadhan, S.: Robust PCPs of proximity, shorter PCPs, and applications to coding. SIAM Journal on Computing 36(4), 889–974 (2006)
23. Ben-Sasson, E., Goldreich, O., Harsha, P., Sudan, M., Vadhan, S.: Robust PCPs of proximity, shorter PCPs, and applications to coding. SIAM Journal on Computing 36(4), 889–974 (2006)
24. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012 (2012)
25. Bitansky, N., Chiesa, A., Ishai, Y., Paneth, O., Ostrovsky, R.: Succinct non-interactive arguments via linear interactive proofs. In: Theory of Cryptography (2013)
26. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA (1988)
27. Boneh, D., Ishai, Y., Sahai, A., Wu, D.J.: Lattice-based SNARGs and their application to more efficient obfuscation. In: EUROCRYPT (2017)

28. Boneh, D., Ishai, Y., Sahai, A., Wu, D.J.: Quasi-optimal SNARGs via linear multi-prover interactive proofs. In: EUROCRYPT (2018)
29. Bootle, J., Cerulli, A., Chaidos, P., Groth, J., Petit, C.: Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In: EUROCRYPT (2016)
30. Bootle, J., Cerulli, A., Ghadafi, E., Groth, J., Hajiabadi, M., Jakobsen, S.K.: Linear-time zero-knowledge proofs for arithmetic circuit satisfiability. In: ASIACRYPT (2017)
31. Bootle, J., Groth, J.: Efficient batch zero-knowledge arguments for low degree polynomials. In: Proc. of Public-Key Cryptography (2018)
32. Boyle, E., Gilboa, N., Ishai, Y.: Function secret sharing: Improvements and extensions. In: CCS (2016)
33. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Efficient range proofs for confidential transactions. Cryptology ePrint Archive, Report 2017/1066 (2017)
34. Catalano, D., Fiore, D.: Vector commitments and their applications. In: PKC (2013)
35. Chakrabarti, A., Cormode, G., McGregor, A., Thaler, J.: Annotations in data streams. ACM Transactions on Algorithms 11(1), 7 (2014)
36. Chida, K., Genkin, D., Hamada, K., Ikarashi, D., Kikuchi, R., Lindell, Y., Nof, A.: Fast large-scale honest-majority MPC for malicious adversaries. In: CRYPTO (2018)
37. Cormode, G., Mitzenmacher, M., Thaler, J.: Practical verified computation with streaming interactive proofs. In: ITCS (2012)
38. Cormode, G., Thaler, J., Yi, K.: Verifying computations with streaming interactive proofs. Proceedings of the VLDB Endowment 5(1), 25–36 (2011)
39. Corrigan-Gibbs, H., Boneh, D.: Prio: Private, robust, and scalable computation of aggregate statistics. In: NSDI (2017)
40. Corrigan-Gibbs, H., Boneh, D., Mazières, D.: Riposte: An anonymous messaging system handling millions of users. In: Symposium on Security and Privacy (2015)
41. Costello, C., Fournet, C., Howell, J., Kohlweiss, M., Kreuter, B., Naehrig, M., Parno, B., Zahur, S.: Geppetto: Versatile verifiable computation. In: 2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015 (2015)
42. Couteau, G.: A note on the communication complexity of multiparty computation in the correlated randomness model. IACR Cryptology ePrint Archive 2018, 465 (2018)
43. Damgård, I., Luo, J., Oechsner, S., Scholl, P., Simkin, M.: Compact zero-knowledge proofs of small Hamming weight. In: PKC (2018)
44. Damgård, I., Nielsen, J.B.: Scalable and unconditionally secure multiparty computation. In: CRYPTO 2007 (2007)
45. Damgård, I., Nielsen, J.B., Nielsen, M., Ranellucci, S.: The tinytable protocol for 2-party secure computation, or: Gate-scrambling revisited. In: CRYPTO (2017)
46. Eerikson, H., Orlandi, C., Pullonen, P., Puura, J., Simkin, M.: Use your brain! arithmetic 3pc for any modulus with active security. IACR Cryptology ePrint Archive 2019, 164 (2019)
47. Feige, U., Goldwasser, S., Lovász, L., Safra, S., Szegedy, M.: Approximating clique is almost NP-complete. In: FOCS (1991)
48. Fortnow, L., Rompel, J., Sipser, M.: On the power of multi-prover interactive protocols. Theoretical Computer Science 134(2), 545–557 (1994)
49. Fortnow, L., Santhanam, R.: Infeasibility of instance compression and succinct PCPs for NP. In: STOC (2008)
50. Furukawa, J., Lindell, Y., Nof, A., Weinstein, O.: High-throughput secure three-party computation for malicious adversaries and an honest majority. In: EUROCRYPT (2017)
51. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct NIZKs without PCPs. In: CRYPTO (2013)
52. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: STOC (1987)
53. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: interactive proofs for muggles. In: STOC (2008)
54. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: Interactive proofs for Muggles. Journal of the ACM 62(4), 27 (2015)
55. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing 18(1), 186–208 (1989)
56. Gordon, S.D., Ranellucci, S., Wang, X.: Secure computation with low communication from cross-checking. IACR Cryptology ePrint Archive 2018, 216 (2018)
57. Groth, J.: Short pairing-based non-interactive zero-knowledge arguments. In: Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings (2010)
58. Groth, J.: On the size of pairing-based non-interactive arguments. In: EUROCRYPT (2016)
59. Gur, T., Rothblum, R.D.: A hierarchy theorem for interactive proofs of proximity. In: 8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA (2017)
60. Ishai, Y., Kushilevitz, E., Ostrovsky, R.: Efficient arguments without short PCPs. In: Conference on Computational Complexity (2007)
61. Kalai, Y.T., Raz, R.: Interactive PCP. In: ICALP (2008)
62. Kalai, Y.T., Rothblum, R.D.: Arguments of proximity. In: CRYPTO (2015)

63. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: ASIACRYPT (2010)
64. Katz, J., Kolesnikov, V., Wang, X.: Improved non-interactive zero knowledge with applications to post-quantum signatures. Tech. rep., Cryptology ePrint Archive, Report 2018/475 (2018)
65. Kilian, J.: A note on efficient zero-knowledge proofs and arguments. In: STOC (1992)
66. Klauck, H.: Rectangle size bounds and threshold covers in communication complexity. In: Conference on Computational Complexity (2003)
67. Kol, G., Oshman, R., Saxena, R.R.: Interactive distributed proofs. In: Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, Egham, United Kingdom, July 23-27, 2018 (2018)
68. Kushilevitz, E.: Communication complexity. In: Advances in Computers, vol. 44 (1997)
69. Lindell, Y., Nof, A.: A framework for constructing fast MPC over arithmetic circuits with malicious adversaries and an honest-majority. In: ACM SIGSAC Conference on Computer and Communications Security, CCS (2017)
70. Lipmaa, H.: Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In: Theory of Cryptography Conference (2012)
71. Micali, S.: CS proofs. In: FOCS (1994)
72. Mohassel, P., Rosulek, M., Zhang, Y.: Fast and secure three-party computation: The garbled circuit approach. In: ACM SIGSAC Conference on Computer and Communications Security, CCS (2015)
73. Naor, M., Parter, M., Yogev, E.: The power of distributed verifiers in interactive proofs. https://arxiv.org/abs/1812.10917 (2018)
74. Nordholt, P.S., Veeningen, M.: Minimising communication in honest-majority MPC by batchwise multiplication verification. In: ACNS (2018)
75. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: EUROCRYPT (1999)
76. Parno, B., Howell, J., Gentry, C., Raykova, M.: Pinocchio: Nearly practical verifiable computation. In: Symposium on Security and Privacy (2013)
77. Parno, B., Howell, J., Gentry, C., Raykova, M.: Pinocchio: nearly practical verifiable computation. Commun. ACM 59(2), 103–112 (2016)
78. Reingold, O., Rothblum, G.N., Rothblum, R.D.: Constant-round interactive proofs for delegating computation. In: Proceedings of the forty-eighth annual ACM symposium on Theory of Computing (2016)
79. Rothblum, G.N., Vadhan, S.P.: Are PCPs inherent in efficient arguments? Computational Complexity 19(2), 265–304 (2010)
80. Sasson, E.B., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized anonymous payments from Bitcoin. In: Symposium on Security and Privacy (2014)
81. Setty, S., Braun, B., Vu, V., Blumberg, A.J., Parno, B., Walfish, M.: Resolving the conflict between generality and plausibility in verified computation. In: EuroSys (2013)
82. Setty, S.T., McPherson, R., Blumberg, A.J., Walfish, M.: Making argument systems for outsourced computation practical (sometimes). In: NDSS (2012)
83. Shamir, A.: IP = PSPACE. Journal of the ACM 39(4), 869–877 (1992)
84. Sudan, M.: Probabilistically checkable proofs. Communications of the ACM 52(3), 76–84 (2009)
85. Wahby, R.S., Tzialla, I., shelat, a., Thaler, J., Walfish, M.: Doubly-efficient zkSNARKs without trusted setup (2018)
86. Williams, R.: Strong ETH breaks with Merlin and Arthur: Short non-interactive proofs of batch evaluation. arXiv preprint arXiv:1601.04743 (2016)
87. Zhang, Y., Genkin, D., Katz, J., Papadopoulos, D., Papamanthou, C.: vSQL: Verifying arbitrary SQL queries over dynamic outsourced databases. In: Symposium on Security and Privacy (2017)
88. Zhang, Y., Genkin, D., Katz, J., Papadopoulos, D., Papamanthou, C.: A zero-knowledge version of vSQL. Cryptology ePrint Archive, Report 2017/1146 (2017)