Revisiting Sum-check-based Polynomial Commitment Schemes

Yuncong Zhang[1]

School of Cyber Science and Technology, Shandong University
yuncong@sdu.edu.cn

Abstract. The sum-check protocol serves as a fundamental building block in succinct arguments, yet its security guarantees are typically formalized in a manner intimately tied to the encompassing protocol, impeding modular analysis and design. We address this limitation by introducing a functional perspective that characterizes sum-check via the *repulsive verifier property*: the verifier's output function is repelled from any polynomial whose hypercube sum differs from the claimed value. This property is intrinsic to sum-check and holds independently of any external binding mechanism or commitment scheme.
Dually, we identify an *attractive verifier property* inherent to the FRI protocol: provided the input word is sufficiently close to a codeword (i.e., a Reed-Solomon encoding of a polynomial), the verifier either outputs the evaluation of that polynomial at the random challenge point or aborts. These dual properties—repulsiveness and attractiveness—yield a clean, modular methodology for analyzing protocols that interleave sum-check with other components.
We demonstrate the efficacy of this methodology through three applications. First, we provide a modular analysis of BaseFold that directly exploits these functional properties, circumventing the need for monolithic security arguments. Second, we show that BulletProofs can be decomposed into a repulsive sum-check and a computationally attractive component, leading to a simpler security proof. Third, we construct Bird, a new polynomial commitment scheme that simultaneously achieves transparency, homomorphic commitments, and double efficiency by composing these functional building blocks, thereby illustrating that our approach facilitates the systematic design of novel cryptographic protocols.

Keywords: Proof System, SNARK, Sum-check, Polynomial Commitment Scheme

1 Introduction

The sum-check protocol, introduced by Lund et al. [29], constitutes a cornerstone of modern cryptographic proof systems. Originally formulated to verify claims regarding the sum of a multivariate polynomial over a Boolean hypercube, it has evolved into a fundamental building block for succinct arguments [19], particularly in the construction of Succinct Non-interactive Arguments of Knowledge

(SNARKs) [24, 37, 43]. Its versatility and efficiency have spurred extensive research into optimizing its performance [4, 5] tailored for SNARK applications.

Recently, a significant paradigm shift has occurred in the utilization of sum-check. While early systems deployed sum-check alongside independent polynomial commitment schemes (PCS) [1,46], modern constructions increasingly leverage sum-check *internally* to instantiate more efficient PCSs. Two prominent examples illustrate this trend: BaseFold [45] achieves $O(n)$ prover complexity and $O(\log^2 n)$ verifier complexity by interleaving sum-check with the Fast Reed-Solomon IOP of Proximity (FRI) protocol [6]. Similarly, BulletProofs [11], a widely adopted inner-product argument, was later revealed by Bootle et al. [10] to fundamentally instantiate a sum-check protocol.

A common architectural pattern unifies these constructions: sum-check is not employed in isolation but operates in tandem with a distinct mechanism responsible for polynomial evaluation. In BaseFold, sum-check is paired with FRI; in BulletProofs, it is paired with a group-based evaluation mechanism. This recurring structure suggests that the synergy between sum-check and evaluation protocols is not coincidental but rather indicative of a deeper design principle. A unified understanding of this interaction could facilitate the modular analysis of such complex systems and, more importantly, enable the synthesis of novel protocols by interchanging compatible components.

However, formalizing this unification presents a subtle challenge. The standard security definition of sum-check is formulated relative to a polynomial oracle, thereby treating the evaluation check as an external assumption. This abstraction introduces a disconnect when analyzing protocols like BaseFold and BulletProofs, wherein the "oracle" is replaced by a dynamic interactive evaluation protocol. The rigidity of the oracle model obscures the intrinsic interaction between the reduction logic of the sum-check protocol and the binding properties of the evaluation protocol, rendering it difficult to capture modular composition using traditional definitions.

To address this limitation and fully capture the potential of these compositions, we introduce a functional perspective on interactive proofs. Rather than defining security via oracle access, we characterize protocols by the intrinsic properties of the function computed by the verifier—what we term the *verifier function*. From this viewpoint, the sum-check protocol is defined by a *repulsive* property: its verifier function is repelled by any polynomial that sums to an incorrect value. Conversely, evaluation protocols like FRI and the core of BulletProofs are defined by an *attractive* property: their verifier functions are inextricably attracted to the evaluation of a specific, committed polynomial.

This functional duality—repulsion versus attraction—provides a powerful lens for modular analysis. The security of complex systems like BaseFold and BulletProofs can be understood simply as the intersection of these two properties: if the sum-check component repels all polynomials with an incorrect sum, and the evaluation component attracts the output to a fixed polynomial, then the only way for their outputs to coincide is if the fixed polynomial yields the claimed sum. Guided by this insight, we not only provide simplified and modular

security proofs for BaseFold and BulletProofs but also synthesize a new polynomial commitment scheme, which we call Bird. By interleaving BulletProofs with a group-valued variant of the efficient FRI evaluation protocol, Bird simultaneously achieves transparency, doubly efficient performance (linear prover time, logarithmic verifier time), and homomorphic commitments.

1.1 Our Contributions

We formalize these insights through three main contributions.

A functional method for dynamic security properties. We propose a novel methodology for analyzing interactive proofs that formalizes security guarantees eluding standard soundness definitions. Whereas classical soundness notions address static statements fixed prior to protocol execution, contemporary protocols increasingly depend on a dynamic property: the ability to evaluate a function on a randomly generated challenge that emerges during execution. Our method formalizes this perspective by modeling the verifier's output as a function of the protocol's random challenges. Through this functional lens, we characterize the sum-check protocol as exhibiting a *repulsive* verifier: the verifier's output function is automatically "repelled" from *any* polynomial with an incorrect sum, coinciding with the polynomial's true evaluation with at most negligible probability. Conversely, we characterize evaluation protocols such as FRI as exhibiting an *attractive* verifier: the verifier's output is "attracted" to a specific committed polynomial, constraining the output to match the polynomial's evaluation on the random challenge. This dichotomy precisely captures the evasive security guarantee of sum-check and the polynomial random evaluation functionality of FRI in a self-contained, composable manner.

Modular analysis of BaseFold and BulletProofs. We employ our functional methodology to streamline the security analysis of prominent polynomial commitment constructions. In the first case, we demonstrate that BaseFold [45] admits an interpretation as the parallel composition of a repulsive sum-check subprotocol and an attractive FRI subprotocol. Its soundness then follows directly: an incorrect claim compels the adversary to output a value that must simultaneously satisfy the repulsive property, which forbids any falsified evaluation, and the attractive property, which enforces the correct evaluation. In the second case, we decompose BulletProofs [11] analogously, into a repulsive sum-check component and a group-valued polynomial random evaluation subprotocol that we term BulletEval. We establish that BulletEval satisfies a computational analogue of the attractive property under the Discrete Logarithm assumption, thereby yielding a modular proof of knowledge soundness for the entire system.

Bird: a new doubly efficient, transparent and homomorphic polynomial commitment scheme. We propose Bird, a novel polynomial commitment scheme that simultaneously achieves transparency, homomorphic commitments, and doubly efficient complexity (with $O(n)$ prover and $O(\mu \cdot q)$ verifier complexity, where $\mu = \log n$ and q is the number of queries). The central observation is

that the primary verifier bottleneck in BulletProofs arises from the direct evaluation of a group-valued polynomial at a random point, which requires linear-time computation for the verifier. This bottleneck can be eliminated by transforming that evaluation into an explicitly delegated subprotocol. Concretely, inspired by BaseFold, our method interleaves BulletProofs with a group-valued FRI evaluation protocol, thereby relieving the verifier from performing the full linear-time group evaluation locally. Such an interleaved protocol would conventionally necessitate a new monolithic security argument for the entire construction, whereas our method enables a modular security proof through the composition of the functional properties of the constituent protocols. Although Eagen et al. [15] also address the same bottleneck by delegating the group-valued evaluation, their design employs a sequential composition (BulletProofs followed by a group-valued variant of BaseFold), while Bird adopts an interleaved group-valued FRI design that reduces concrete overhead compared to that sequential approach.

1.2 Technical Overview

Formalizing security of sum-check. In its high-level description, the sum-check protocol constitutes an interactive proof for the claim

$$\sum_{\mathbf{b}\in\{0,1\}^{\mu}} f(\mathbf{b}) = c,$$

where $f(\mathbf{X})$ denotes a μ-variate polynomial and $c \in \mathbb{F}$ is the claimed sum. During the interaction, the verifier $\mathcal{V}$ sequentially samples challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$ from $\mathbb{F}^{\mu}$ across μ rounds, and upon termination, outputs a value y intended to equal $f(\mathbf{r})$. The standard soundness guarantee dictates that if $\sum_{\mathbf{b}\in\{0,1\}^{\mu}} f(\mathbf{b}) \neq c$, then for any cheating prover $\mathcal{P}^*$,

$$\Pr_{\mathbf{r}}\big[y = f(\mathbf{r})\big] \leq \varepsilon_{\mathrm{sc}},$$

with $\varepsilon_{\mathrm{sc}}$ small. In other words, an incorrect sum claim yields a correct evaluation at a random point only with negligible probability.

Our technical departure point is a fundamental yet surprisingly nuanced question: how can the soundness of sum-check be formalized in a standalone, context-independent fashion? The conventional approach rooted in interactive proofs for NP relations proves awkward in this setting: if the instance comprises solely the claimed sum c, the induced relation is essentially vacuous (a witness being "some polynomial with sum c" is trivial to provide). To render the statement meaningful, one typically augments the model with additional structure—such as a binding access mechanism to a fixed polynomial, e.g., oracle access or a cryptographic commitment—but then soundness ceases to be a property of sum-check alone and becomes contingent upon this auxiliary mechanism.

A common alternative casts sum-check as a reduction from a sum claim to an evaluation claim. Yet this perspective is also unsatisfactory for modular reasoning. In BaseFold, for instance, where sum-check is executed interleaved

with FRI, one cannot exploit this reduction's security guarantee in a black-box manner; consequently, existing analyses of BaseFold must delve into the specifics of the sum-check protocol. Moreover, the reduction implicitly assumes that both claims pertain to the same fixed polynomial throughout the interaction, thereby reintroducing hidden binding assumptions via oracle or commitment machinery.

To isolate the core mathematical essence of sum-check itself, we adopt an alternative vantage point: we treat the verifier's final output y (produced after μ rounds) as a function determined by the complete description of the prover $\mathcal{P}$ (which, without loss of generality, we assume to be deterministic) and the claimed sum c. We denote this function by $\mathcal{V}_{\mathcal{P},c} : \mathbb{F}^\mu \to \mathbb{F}$ and refer to it as the *verifier function*. This perspective reframes the prover's objective from "forcing the verifier to output a particular value" to "shaping the behavior of $\mathcal{V}_{\mathcal{P},c}$ in a specific manner." The security guarantee of sum-check is then captured by answering the following question: what functions can the prover (or cannot) induce the verifier to evaluate?

This inquiry yields a surprisingly succinct characterization of the intrinsic guarantee of sum-check: for any prover $\mathcal{P}$ and sum c, $\mathcal{V}_{\mathcal{P},c}$ must be *far from* every polynomial whose hypercube sum differs from c. Formally, for any $\mathcal{P}$, any c, and any polynomial $f(\mathbf{X})$ with hypercube sum $c' \neq c$, we have

$$\Pr_{\mathbf{r}}\big[\mathcal{V}_{\mathcal{P},c}(\mathbf{r}) = f(\mathbf{r})\big] \leq \varepsilon_{\mathrm{sc}}.$$

Intuitively, we say that the sum-check verifier is *repulsive* to any polynomial f with an incorrect sum.

From repulsive sum-check to attractive FRI. In its conventional role, the Fast Reed–Solomon IOP of Proximity (FRI) serves as a proximity-testing protocol: given oracle access to a word $w \in \mathbb{F}^N$, the verifier determines whether w is close to a Reed–Solomon code $\mathsf{RS}[\mathbb{F}, L, \rho]$ over a domain L with rate ρ. This is equivalent to testing proximity to a low-degree polynomial $f(X) \in \mathbb{F}^d[X]$ evaluated on L, where $d = \rho \cdot |L|$. FRI proceeds in $\mu = \log d$ rounds. In each round, the verifier sends a random challenge $r_i \in \mathbb{F}$ to the prover, who then folds the coefficients of f under r_i and provides the resulting codeword of half the size (again in oracle form) back to the verifier. In the final round, when the polynomial reduces to a constant, the prover transmits this constant y directly to the verifier.[1] The parties subsequently engage in a query phase, during which the verifier queries the oracles provided by the prover at random positions to verify the consistency of the transmitted codewords.

The standard soundness guarantee captures exactly the low-degree testing property: if w is far from the code, the probability of acceptance is small. A crucial observation from BaseFold is that the final output y of FRI is precisely the evaluation of f at $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$, where the coefficients of f are interpreted as a multilinear polynomial rather than a univariate one. This property makes FRI particularly valuable as a building block in BaseFold.

[1] FRI usually terminates earlier, before the polynomial becomes constant, but this simplification does not affect our analysis.

One would naturally expect that FRI guarantees the following: irrespective of the prover's strategy, y equals $f(\mathbf{r})$ whenever the verifier does not abort (i.e., when no cheating is detected). Unfortunately, this guarantee is not directly captured by the standard proximity-only soundness statement. While the dynamic nature of the statement—the claim depends on the randomness $\mathbf{r}$ sampled during protocol execution—makes it challenging to formalize via traditional definitions, our functional perspective on sum-check adapts readily to this setting and yields a more precise security characterization.

Formally, let $\mathcal{V}_{\mathcal{P},[w]} : \mathbb{F}^{\mu} \to \mathbb{F}$ be the function computed by the verifier $\mathcal{V}$ (on input $\mathbf{r} \in \mathbb{F}^{\mu}$ drawn from its random tape) when interacting with the prover $\mathcal{P}$, where $\mathcal{V}$ is granted oracle access to the word $w \in \mathbb{F}^N$. Fix a proximity parameter $\delta < \frac{1-\rho}{2}$ for the FRI protocol. We assert that for any $w \in \mathbb{F}^N$ that is δ-close (in relative Hamming distance) to the Reed–Solomon code $\mathsf{RS}[\mathbb{F}, L, \rho]$, hence uniquely decodeable to a polynomial f, the following holds for any (even computationally unbounded) prover $\mathcal{P}$:

$$\Pr_{\mathbf{r}}\left[\mathcal{V}_{\mathcal{P},[w]}(\mathbf{r}) \in \{f(\mathbf{r}), \perp\}\right] \geq 1 - \varepsilon_{\mathsf{FRI}},$$

where $\perp$ denotes that the verifier aborts and produces no output. Intuitively, the FRI verifier is *attractive* with respect to the polynomial $f(\mathbf{X})$: except with negligible probability, its output on a random challenge is either the correct evaluation $f(\mathbf{r})$ or an abort.

Recent advances in FRI analysis have pushed the proximity parameter δ well beyond the unique-decoding radius $\frac{1-\rho}{2}$ into the *list-decoding regime*, where the word may be decoded to many possible polynomials. To accommodate this scenario, one could generalize the definition of attractiveness from a single function to a set of functions. However, establishing this property for FRI at such thresholds would be intricate and likely as complex as proving its standard soundness. To avoid obscuring the main methodological contribution of this work, we defer this extension to future investigation. Instead, to demonstrate the effectiveness of our functional framework, we prove the above attractive property for a conservative proximity parameter $\delta = (1-\rho)/4$, using a modular reduction to standard FRI soundness; this yields a substantially simpler proof.

Simpler analysis of BaseFold. The functional characterizations of sum-check and FRI introduced above facilitate a modular, and thus considerably simpler, security analysis of the BaseFold protocol. This analysis achieves similar quantitative soundness guarantee as the original BaseFold paper, namely soundness for a proximity parameter at the unique decoding radius. At a high level, BaseFold consists of the parallel execution of (i) a sum-check protocol for the claim $\sum_{\mathbf{b}} f(\mathbf{b}) \cdot \mathsf{eq}(\mathbf{b}, \mathbf{z}) = y$ and (ii) an FRI protocol for the Reed–Solomon encoding of $f(\mathbf{X})$, with both protocols sharing the same random challenges $\mathbf{r}$. The construction exploits the identity that the sum of $f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$ over the Boolean hypercube equals $f(\mathbf{z})$. The critical insight is to employ the final output of the FRI protocol—which, by the attractive property, must equal $f(\mathbf{r})$ unless the verifier aborts—to certify the evaluation claim output by the sum-check subprotocol. Prior to our work, the lack of a modular framework to exploit the security

guarantee of sum-check in an interleaved setting forced the original analysis of BaseFold to rely on a monolithic argument that delves into the internal details of the sum-check protocol.

In contrast, our functional methodology enables a clean, modular proof. At a high level, our analysis leverages the repulsive property of sum-check—which prevents agreement with any polynomial that sums incorrectly—and the attractive property of FRI—which forces the output to coincide with the evaluation of a committed polynomial f. By requiring the two outputs to be consistent, the protocol ensures that f must indeed satisfy the claimed sum. Formally, let y_{sc} denote the output of the sum-check component (instantiated for $f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$ with claimed sum y) and y_{fri} the output of the FRI component; the verifier accepts only if $y_{\mathsf{fri}} \neq \perp$ and $y_{\mathsf{sc}} = y_{\mathsf{fri}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$. Provided that w is δ-close to a codeword that uniquely decodes to f, the attractive property of FRI guarantees $y_{\mathsf{fri}} \in \{f(\mathbf{r}), \perp\}$ with probability at least $1 - \varepsilon_{\mathsf{FRI}}$. Meanwhile, by the repulsive property of sum-check, for any polynomial g satisfying $g(\mathbf{z}) \neq y$, we have $\Pr_{\mathbf{r}}[y_{\mathsf{sc}} = g(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})] \leq \varepsilon_{\mathsf{sc}}$. Consequently, on the high-probability event that the attractive property succeeds and $y_{\mathsf{fri}} \neq \perp$, the acceptance condition forces $y_{\mathsf{sc}} = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$. By the repulsive guarantee, such an equality can occur with non-negligible probability only if $f(\mathbf{z}) = y$. A union bound then yields $\Pr[\mathsf{accept} \wedge f(\mathbf{z}) \neq y] \leq \varepsilon_{\mathsf{sc}} + \varepsilon_{\mathsf{fri}}$.

A crucial observation underpinning this argument is that interleaving the two protocols does not undermine the functional properties of either component. Formally, let $\mathcal{V}_{\mathcal{P},y,[w]} : \mathbb{F}^{\mu} \to \mathbb{F} \times \mathbb{F}$ denote the joint verifier function mapping challenges $\mathbf{r}$ to the pair $(y_{\mathsf{sc}}, y_{\mathsf{fri}})$. Projecting onto the first or second coordinate yields exactly the verifier functions of the individual sum-check and FRI protocols, parameterized respectively by $(\mathcal{P}_{\mathsf{sc}}, y)$ and $(\mathcal{P}_{\mathsf{fri}}, [w])$, where $\mathcal{P}_{\mathsf{sc}}$ and $\mathcal{P}_{\mathsf{fri}}$ are the prover strategies restricted to their respective outputs. Because the repulsive and attractive properties hold against arbitrary prover strategies, they are preserved under concatenation, i.e., when the two sub-protocols are combined.

Simpler analysis of BulletProofs. We extend this modular methodology to the analysis of BulletProofs, now viewed in the polynomial commitment setting. In this setting, a prover commits to a multilinear polynomial $f(\mathbf{X})$ via the group relation $C = \langle \mathbf{f}, \mathbf{G} \rangle$, and later proves an evaluation claim $f(\mathbf{z}) = y$, where $\mathbf{f}$ is the evaluation of f over $\{0,1\}^{\mu}$. Conceptually, BulletProofs executes two coupled sum-check–like protocols under shared randomness $\mathbf{r}$: a scalar-valued sum-check for $f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$ (claiming sum y), and a group-valued sum-check for $f(\mathbf{X}) \cdot \tilde{\mathbf{G}}(\mathbf{X})$ (claiming sum C), where $\tilde{\mathbf{G}}(\mathbf{X})$ is the group-valued polynomial obtained by interpolating $\mathbf{G}$. The latter component thereby binds the former to a committed polynomial.

To transform the group-valued sum-check into a polynomial random evaluation protocol analogous to FRI in BaseFold, we define $\mathsf{BulletEval}$ as follows. After μ rounds, the underlying sum-check yields a folded commitment $C^{(\mu)}$, which in an honest execution satisfies $C^{(\mu)} = \tilde{\mathbf{G}}(\mathbf{r}) \cdot f(\mathbf{r})$. The prover additionally supplies a scalar y_{be}; the verifier checks $C^{(\mu)} = \tilde{\mathbf{G}}(\mathbf{r}) \cdot y_{\mathsf{be}}$ and outputs y_{be} if the check holds, and $\perp$ otherwise. The resulting protocol directly parallels the Base-

Fold construction: the scalar sum-check provides a repulsive guarantee against incorrect evaluation claims, BulletEval supplies a computational attractive guarantee that binds the verifier's output to an extracted committed polynomial, and a consistency check $y_{\mathsf{sc}} = y_{\mathsf{be}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$ enforces agreement between the two outputs. Soundness follows modularly from the composition of these properties, eliminating the need for a monolithic analysis of the entire interleaved transcript.

A key distinction is that BulletEval, despite being constructed from a group-valued sum-check, is not repulsive but *attractive*: the relation $C = \langle \mathbf{f}, \mathbf{G} \rangle$ binds the verifier's output to evaluations of a committed polynomial, rather than merely repelling incorrect sum claims. This gives rise to definitional subtleties absent in the information-theoretic FRI case. First, attractiveness must be formulated computationally: the guarantee no longer holds against unbounded provers, so we restrict to probabilistic polynomial-time adversaries. Second, the relation depends on uniformly random public parameters $\mathbf{G}$; consequently, the security statement should hold with high probability over the choice of $\mathbf{G}$, rather than as a worst-case guarantee for any fixed $\mathbf{G}$. Third, for a given $(C, \mathbf{G})$ there may be many polynomials f satisfying $\langle \mathbf{f}, \mathbf{G} \rangle = C$, so a naïve claim such as "$\mathcal{V}_{\mathcal{P},(C,\mathbf{G})}(\mathbf{r})$ is close to $\varphi((C, \mathbf{G}), f, \mathbf{r})$" is ill-defined without specifying which f is selected.

To address these, we adopt a knowledge-style definition of attractiveness: there exists an extractor $\mathcal{E}$ such that, for any PPT prover strategy that selects C after seeing random $\mathbf{G}$, if the verifier outputs $y \neq \perp$ with non-negligible probability, then $\mathcal{E}$ extracts a polynomial f satisfying $\langle \mathbf{f}, \mathbf{G} \rangle = C$, and, except with negligible probability over the random challenges, $y = f(\mathbf{r})$. This computational analogue of attraction provides precisely the property required for modular composition with the repulsive scalar sum-check.

From BulletProofs to Bird. The above decomposition identifies BulletEval as the dominant verifier bottleneck in BulletProofs: to finalize the group-valued sum-check by verifying $C^{(\mu)} = \tilde{\mathbf{G}}(\mathbf{r}) \cdot y_{\mathsf{be}}$, the verifier must compute $\tilde{\mathbf{G}}(\mathbf{r})$ locally, which incurs a linear number of group operations (with respect to the lengths of $\mathbf{G}$ and f) and dominates the verifier's runtime. Inspired by BaseFold, which delegates the computation of $f(\mathbf{r})$ to the prover via an FRI evaluation subprotocol, our new PCS Bird eliminates this bottleneck by substituting the local evaluation of $\tilde{\mathbf{G}}(\mathbf{r})$ with an interleaved group-valued FRI subprotocol that certifies the evaluation.

For security analysis, traditional monolithic arguments become involved because they must reason over a tightly coupled transcript involving multiple dynamic checks; in contrast, our functional methodology yields a modular proof that composes three independent properties: the repulsiveness of the scalar sum-check, the computational attractiveness of BulletEval, and the attractiveness of the group-valued FRI evaluation. This also clarifies the relationship to the concurrent work of Eagen et al. [15]: they address the same linear verifier bottleneck through a sequential design (BulletProofs followed by a group-valued variant of BaseFold), whereas Bird uses an interleaved design with group-valued FRI, thereby reducing concrete overhead.

1.3 Related Works

Sum-check. The sum-check protocol, introduced by Lund et al. [29], is an interactive proof system for verifying the sum of a multivariate polynomial over a Boolean hypercube. While earlier works are surveyed in the introduction, the most directly relevant contribution is the Sum-check Argument of Bootle et al. [10], which elucidates the connection between BulletProofs and sum-check. In contrast to that work, which interprets BulletProofs as a single sum-check for a three-dimensional polynomial (with coefficients from $\mathbb{G} \times \mathbb{G} \times \mathbb{F}$), we formalize BulletProofs as the parallel composition of BulletEval and a sum-check, thereby simplifying the analysis and enabling more flexible compositions, e.g., pairing BulletEval with protocols other than sum-check.

SNARKs. Succinct non-interactive arguments of knowledge (SNARKs) [8, 33] prove correct execution of a program through short and efficiently verifiable proofs. This technology has experienced rapid development in recent years, driven by its generality and high performance [2, 12–14, 16, 17, 22, 24, 27, 30, 37, 38, 43, 45–47]. SNARKs have found widespread deployment in practical applications, particularly in blockchains for scalability and privacy [3, 28, 36, 48], and hold potential for broader adoption in cloud computing [31, 42] and verifiable machine learning [26, 34, 39, 40].

The increasing complexity of SNARK constructions has motivated the development of modular designs to enhance clarity and reusability. A prominent example is the Polynomial Interactive Oracle Proof (PIOP) pipeline [12], which was concurrently developed under different names including Algebraic Holographic Proof (AHP) [14] and polynomial protocols [17], and has become the predominant methodology for SNARK construction. Further modularization efforts include the work by Ràfols et al. [35], which formalizes Checkable Subspace Sampling Arguments for pairing-based SNARKs, and Zhang et al. [47], who introduce the Vector Oracle model to further decompose PIOP constructions.

Polynomial commitment schemes. As a central component of the PIOP pipeline, PCS [21] have attracted extensive research attention, leading to numerous proposed constructions [1,2,6,7,12,18,20,25,44–46]. The pursuit of transparent PCS (without a trusted setup) has been particularly fruitful, with early constructions such as Hyrax [41] demonstrating the feasibility of transparent arguments for data-parallel statements, and Virgo [46] achieving transparent polynomial delegation with sublinear verification. Recent PCS research has shifted focus from early univariate polynomials [6, 21] toward multilinear polynomials, motivated by the goal of SNARKs with linear-time prover complexity. This has spurred dedicated multilinear PCS constructions [1, 18, 23, 45, 46] as well as transformation techniques that adapt univariate schemes for multilinear use [9, 32].

Concurrent to our work, Eagen et al. [15] proposed a PCS that combines BulletProofs with a group-valued variant of BaseFold, achieving asymptotic efficiency comparable to our construction. However, our approach offers several advantages: we employ parallel rather than sequential composition, which reduces the round complexity by half, and we directly compose BulletProofs with

FRI rather than with BaseFold, eliminating the additional sum-check incurred by BaseFold.

2 Preliminaries

Let $\mathbb{F}$ be a finite field and $\mathbb{G}$ be a cyclic group of prime order p. We denote by λ the security parameter. A function $\mathrm{negl}(\cdot)$ is negligible if for every positive polynomial $p(\cdot)$, there exists $N \in \mathbb{N}$ such that for all $n > N$, $\mathrm{negl}(n) < 1/p(n)$. For a finite set S, we write $x \xleftarrow{\$} S$ to denote sampling x uniformly at random from S. For two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{F}^n$, the *relative Hamming distance* $\Delta(\mathbf{u}, \mathbf{v})$ is the fraction of positions where they differ: $\Delta(\mathbf{u}, \mathbf{v}) = |\{i : u_i \neq v_i\}|/n$. We say $\mathbf{u}$ is δ-*close* to $\mathbf{v}$ if $\Delta(\mathbf{u}, \mathbf{v}) \leq \delta$, and δ-*far* from $\mathbf{v}$ otherwise. For a code $C \subseteq \mathbb{F}^N$, we write $\Delta(\mathbf{u}, C) = \min_{\mathbf{c} \in C} \Delta(\mathbf{u}, \mathbf{c})$ for the distance from $\mathbf{u}$ to the nearest codeword. We write $[\mathbf{w}]$ to denote that a party has oracle access to $\mathbf{w}$, allowing queries to any entry of $\mathbf{w}$ without revealing the entire vector. We write $\perp$ to denote the verifier's rejection or abort signal.

2.1 Interactive Proofs and Arguments

Definition 1 (Interactive Proof System). *An interactive proof system for a language L is a pair of algorithms $(\mathcal{P}, \mathcal{V})$, where $\mathcal{P}$ is a computationally unbounded prover and $\mathcal{V}$ is a probabilistic polynomial-time verifier. For a common input x, the prover holds a private witness w for certifying $x \in L$, and engages in an interactive protocol with the verifier. We denote the interaction and its output by $b \leftarrow \langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle$, where $b \in \{0, 1\}$ indicates acceptance or rejection.*

The system must satisfy:

- ***Completeness:*** *For any $x \in L$ with valid witness w:*

$$\Pr[b = 1 \mid b \leftarrow \langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle] \geq 1 - \varepsilon_c(\lambda),$$

  where $\varepsilon_c(\lambda)$ is the completeness error.
- ***Soundness:*** *For any $x \notin L$ and any (possibly unbounded) prover $\mathcal{P}^*$:*

$$\Pr[b = 1 \mid b \leftarrow \langle \mathcal{P}^*(x), \mathcal{V}(x) \rangle] \leq \varepsilon_s(\lambda),$$

  where $\varepsilon_s(\lambda)$ is the soundness error.

An *argument* (or computationally sound proof) relaxes the soundness requirement so that it holds only against probabilistic polynomial-time provers.

Knowledge soundness. For arguments, a stronger notion called *knowledge soundness* is often required. We adopt the definition based on individual extractability: for any prover that succeeds with non-negligible probability, there exists an extractor that can recover the witness.

Definition 2 (Argument of Knowledge). *An interactive argument $(\mathcal{P}, \mathcal{V})$ for a relation $\mathcal{R}$ is an argument of knowledge with knowledge error κ if there exists an expected polynomial-time extractor $\mathcal{E}$ such that for any polynomial-time prover $\mathcal{P}^*$:*

$$\Pr\left[b = 1 \wedge (x, w) \notin \mathcal{R} \;\middle|\; \begin{array}{l} b \leftarrow \langle \mathcal{P}^*(x), \mathcal{V}(x) \rangle \\ w \leftarrow \mathcal{E}^{\mathsf{nextmsg}[\mathcal{P}^*]}(x) \end{array}\right] \leq \kappa(\lambda) + \mathit{negl}(\lambda),$$

where $\mathcal{E}$ has oracle access to $\mathcal{P}^*$'s next-message function (enabling rewinding), and the probability is taken over the random coins of both parties. The knowledge error κ is typically set to the protocol's soundness error.

The definition quantifies over randomized provers; any randomized prover can be derandomized by fixing its randomness to the value that maximizes its success probability, so it is without loss of generality to restrict to deterministic provers when proving soundness.

2.2 Multilinear Polynomials

For a positive integer μ, we denote by $\{0,1\}^\mu$ the boolean hypercube $\{0,1\}^\mu$. A μ-variate polynomial $f(\mathbf{X}) = f(X_1, \ldots, X_\mu)$ over $\mathbb{F}$ is said to be *multilinear* if each variable appears with degree at most 1.

For any function $f : \{0,1\}^\mu \to \mathbb{F}$, there exists a unique multilinear polynomial $\tilde{f}(\mathbf{X})$ such that $\tilde{f}(\mathbf{b}) = f(\mathbf{b})$ for all $\mathbf{b} \in \{0,1\}^\mu$. We call $\tilde{f}$ the *multilinear extension* of f. For any vector $\mathbf{v}$ of length 2^μ, we identify $\mathbf{v}$ with the function $\mathbf{v}(\mathbf{b})$ for $\mathbf{b} \in \{0,1\}^\mu$ that maps $\mathbf{b}$ to v_i, where i is the integer whose little-endian binary representation is $\mathbf{b}$. Similarly, let $\tilde{\mathbf{v}}$ be the multilinear extension of $\mathbf{v}$.

For $\mathbf{z} \in \mathbb{F}^\mu$, we define the multilinear polynomial

$$\mathsf{eq}(\mathbf{X}, \mathbf{z}) = \prod_{i=1}^{\mu} \big((1 - X_i)(1 - z_i) + X_i z_i\big).$$

This function has the property that for all $\mathbf{b} \in \{0,1\}^\mu$: $\mathsf{eq}(\mathbf{b}, \mathbf{z}) = 1$ if $\mathbf{b} = \mathbf{z}$, and $\mathsf{eq}(\mathbf{b}, \mathbf{z}) = 0$ if $\mathbf{b} \neq \mathbf{z}$. Note that this property holds only when both $\mathbf{b}$ and $\mathbf{z}$ are restricted to the boolean hypercube. A key identity is:

$$\sum_{\mathbf{b} \in \{0,1\}^\mu} f(\mathbf{b}) \cdot \mathsf{eq}(\mathbf{b}, \mathbf{z}) = f(\mathbf{z}).$$

2.3 Polynomial Commitment Schemes

Definition 3 (Polynomial Commitment Scheme). *A polynomial commitment scheme (PCS) for multilinear polynomials consists of algorithms* (Setup, Commit, Open, Eval) *with the following functionality:*

- $\mathsf{Setup}(1^\lambda, D) \to \mathsf{pp}$*: generates public parameters given security parameter λ and polynomial size bound $D = 2^\mu$. The public parameters* pp *may be split into prover parameters and verifier parameters* $\mathsf{vp} \subseteq \mathsf{pp}$ *containing only the information needed for verification.*

- $\mathsf{Commit}(\mathsf{pp}, f) \to \mathsf{cm}$*: produces a commitment* cm *to multilinear polynomial* f *within the size bound* D*.*
- $\mathsf{Open}(\mathsf{pp}, \mathsf{cm}, f) \to b$*: verifies that* f *is a valid opening of* cm*.*
- $\mathsf{Eval}(\mathsf{vp}, \mathsf{cm}, \mathbf{z}, y; f, \mathsf{pp}) \to b$*: an interactive protocol where the prover, given* f *and* pp*, convinces the verifier, given* $\mathsf{vp}, \mathsf{cm}, \mathbf{z}, y$*, that* $f(\mathbf{z}) = y$ *and* $\mathsf{Open}(\mathsf{pp}, \mathsf{cm}, f) = 1$*.*

The scheme must satisfy:

Correctness: For any polynomial f within the size bound:

$$\Pr\left[\mathsf{Open}(\mathsf{pp}, \mathsf{cm}, f) = 1 \;\middle|\; \begin{array}{l} \mathsf{Setup}(1^\lambda, D) \to \mathsf{pp} \\ \mathsf{Commit}(\mathsf{pp}, f) \to \mathsf{cm} \end{array}\right] = 1.$$

Completeness: For any polynomial f within the size bound and any $\mathbf{z}$:

$$\Pr\left[\mathsf{Eval}(\mathsf{vp}, \mathsf{cm}, \mathbf{z}, f(\mathbf{z}); f, \mathsf{pp}) = 1 \;\middle|\; \begin{array}{l} \mathsf{Setup}(1^\lambda, D) \to \mathsf{pp} \\ \mathsf{Commit}(\mathsf{pp}, f) \to \mathsf{cm} \end{array}\right] = 1.$$

Binding: No efficient adversary can produce two distinct polynomials that open to the same commitment:

$$\Pr\left[b_0 = b_1 = 1 \wedge f_0 \neq f_1 \;\middle|\; \begin{array}{l} \mathsf{Setup}(1^\lambda, D) \to \mathsf{pp} \\ \mathcal{A}(\mathsf{pp}) \to (\mathsf{cm}, f_0, f_1) \\ \mathsf{Open}(\mathsf{pp}, \mathsf{cm}, f_i) \to b_i \end{array}\right] \leq \mathrm{negl}(\lambda).$$

Knowledge soundness: The evaluation protocol Eval is an interactive argument of knowledge for the relation

$$\mathcal{R}_{\mathsf{Eval}} = \{((\mathsf{pp}, \mathsf{cm}, \mathbf{z}, y), f) \mid f(\mathbf{z}) = y \wedge \mathsf{Open}(\mathsf{pp}, \mathsf{cm}, f) = 1\}.$$

Definition 4 (Homomorphic PCS). *A PCS is homomorphic if there exists an efficient algorithm* $\mathsf{HomomorphicAdd}$ *such that for any polynomials* f_1, f_2*:*

$$\mathsf{HomomorphicAdd}(\mathsf{Commit}(\mathsf{pp}, f_1), \mathsf{Commit}(\mathsf{pp}, f_2)) = \mathsf{Commit}(\mathsf{pp}, f_1 + f_2).$$

A PCS is *transparent* if Setup requires no trusted setup.

2.4 The Discrete Logarithm Assumptions

Let $\mathbb{G}$ be a cyclic group of prime order p with generator G. We use additive notation for group operations: for $x \in \mathbb{Z}_p$ and $H \in \mathbb{G}$, we write $x \cdot H$ to denote scalar multiplication. The *discrete logarithm* assumption states that given random $H \in \mathbb{G}$, computing x such that $H = x \cdot G$ is computationally infeasible.

The *q-discrete logarithm* (q-DLOG) assumption states: given q uniformly random generators $G_1, G_2, \ldots, G_q \in \mathbb{G}$, it is infeasible to find coefficients $c_1, c_2, \ldots, c_q \in \mathbb{Z}_p$, not all zero, such that $c_1 \cdot G_1 + c_2 \cdot G_2 + \cdots + c_q \cdot G_q = O$, where O is the group identity. This assumption underlies many inner-product arguments, including BulletProofs.

2.5 The Sum-check Protocol

The sum-check protocol is an interactive proof for verifying the sum of a multilinear polynomial over the boolean hypercube. Let $f(\mathbf{X})$ be a μ-variate multilinear polynomial over $\mathbb{F}$ with $n = 2^\mu$ coefficients. The protocol verifies the claim

$$\sum_{\mathbf{b}\in\{0,1\}^\mu} f(\mathbf{b}) = c.$$

Protocol 1 (Sum-Check). Let $f(\mathbf{X})$ be a μ-variate multilinear polynomial over $\mathbb{F}$ with $n = 2^\mu$ coefficients.
Prover input: Multilinear polynomial $f(\mathbf{X})$.
Verifier input: Claimed sum $c \in \mathbb{F}$ and evaluation access to f.

For $i = 0, \ldots, \mu - 1$:

1. The prover sends a univariate polynomial $g_i(X_i)$ claimed to equal
$$\sum_{\mathbf{b}\in\{0,1\}^{\mu-i-1}} f(r_0, \ldots, r_{i-1}, X_i, \mathbf{b}).$$
2. The verifier checks $g_i(0) + g_i(1) = v_i$ (where v_0 is c).
3. The verifier uniformly samples $r_i \xleftarrow{\$} \mathbb{F}$ and sends r_i to the prover.
4. Let $v_{i+1} = g_i(r_i)$.

After μ rounds, the verifier computes $f(r_0, \ldots, r_{\mu-1})$ and checks $v_\mu = f(r_0, \ldots, r_{\mu-1})$. The verifier accepts if all checks pass.

Theorem 1 (Sum-Check Soundness [29]). *For any prover $\mathcal{P}^*$, any claimed sum c, and any multivariate polynomial $f(\mathbf{X}) \in \mathbb{F}^{\leq d}[X_0, \cdots, X_{\mu-1}]$ with*

$$\sum_{\mathbf{b}\in\{0,1\}^\mu} f(\mathbf{b}) \neq c,$$

we have

$$\Pr\big[1 \leftarrow \langle \mathcal{P}^*(c), \mathcal{V}(c, [f])\rangle\big] \leq \frac{d \cdot \mu}{|\mathbb{F}|}.$$

2.6 Inner Product Arguments

Inner product arguments (IPAs) enable a prover to demonstrate knowledge of a committed vector satisfying a specified inner product relation. The most prominent example is BulletProofs [11]. For simplicity, we present only the variant of IPA where one vector is committed and the other is public, which suits the context of polynomial commitment schemes.

Definition 5 (Inner Product Argument). *An inner product argument scheme consists of algorithms* $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Eval})$*:*

- $\mathsf{Setup}(1^\lambda, n) \to \mathsf{pp}$*: generates public parameters for vectors of dimension* n*.*
- $\mathsf{Commit}(\mathsf{pp}, \mathbf{v}) \to \mathsf{cm}$*: produces a commitment to the vector* $\mathbf{v}$*.*
- $\mathsf{Eval}(\mathsf{pp}, \mathsf{cm}, \mathbf{u}, t; \mathbf{v}) \to b$*: an interactive protocol where both parties receive as input the public parameters* pp*, the commitment* cm*, the public vector* $\mathbf{u}$*, and a target* t*. The prover additionally has access to the committed vector* $\mathbf{v}$*. At the end of the protocol, the verifier outputs a bit* b *indicating acceptance or rejection.*

The scheme must satisfy completeness and knowledge soundness for the relation $\mathcal{R}_{\mathsf{Eval}} = \{((\mathsf{pp}, \mathsf{cm}, \mathbf{u}, t), \mathbf{v}) \mid \langle \mathbf{v}, \mathbf{u} \rangle = t \wedge \mathsf{Commit}(\mathsf{pp}, \mathbf{v}) = \mathsf{cm}\}$.

2.7 The FRI Protocol

The Fast Reed-Solomon IOP of Proximity (FRI) [6] is an interactive oracle proof for testing whether a given word is close to a Reed-Solomon code. We present FRI at a high level, providing sufficient detail for understanding this paper.

Notations. We introduce some additional notations specific for describing FRI. Let $\mathbf{v} \in \mathbb{F}^n$ be a message vector where $n = 2^\mu$. Let $\hat{f}_{\mathbf{v}}$ be the univariate polynomial of degree less than n with coefficients given by $\mathbf{v}$. Let $\tilde{f}_{\mathbf{v}}$ be the multilinear polynomial of μ variables with coefficients (ordered in little endian) given by $\mathbf{v}$.

Reed-Solomon codes. For a domain $L \subset \mathbb{F}$ of size N and rate $\rho \in (0, 1)$, the Reed-Solomon code $\mathsf{RS}[\mathbb{F}, L, \rho]$ is the set of evaluations of polynomials of degree less than $n = \rho \cdot N$ over L. For a vector $\mathbf{v} \in \mathbb{F}^n$, define $\mathsf{Encode}_C(\mathbf{v}) = \hat{f}_{\mathbf{v}}(L)$.

Foldable code family. FRI operates on a *foldable family* of Reed-Solomon codes $C^{(0)}, \ldots, C^{(\mu)}$ where each $C^{(i)} = \mathsf{RS}[\mathbb{F}, L^{(i)}, \rho]$ with $|L^{(i)}| = N^{(i)} = |L^{(i-1)}|/2$ and message sizes $n^{(i)} = 2^{\mu - i}$. The key operation is *folding*, which operates on both the message vector $\mathbf{v}$ and the codeword $\mathbf{c}$. Given any $r \in \mathbb{F}$:

- $\mathsf{Fold}(r, \cdot) : C^{(i-1)} \to C^{(i)}$ maps codeword $\mathbf{c} \in \mathbb{F}^{N^{(i-1)}}$ to codeword $\mathbf{c} \in \mathbb{F}^{N^{(i)}}$.
- $\mathsf{Fold}(r, \cdot) : \mathbb{F}^{n^{(i-1)}} \to \mathbb{F}^{n^{(i)}}$ maps message $\mathbf{v} \in \mathbb{F}^{n^{(i-1)}}$ to message $\mathbf{v} \in \mathbb{F}^{n^{(i)}}$.

FRI relies on the following crucial property: folding commutes with encoding. In other words, folding the message is equivalent to folding its encoding. Formally, for any $i > 0$, $r \in \mathbb{F}$ and any $\mathbf{v} \in \mathbb{F}^{n^{(i-1)}}$,

$$\mathsf{Fold}(\mathsf{Encode}_{C^{(i-1)}}(\mathbf{v}), r) = \mathsf{Encode}_{C^{(i)}}(\mathsf{Fold}(\mathbf{v}, r)).$$

Another useful property of the foldable code family, which is exploited by BaseFold, is the equivalence between folding the message $\mathbf{v}$ and evaluating the multilinear polynomial $\tilde{f}_{\mathbf{v}}$. Formally, for any $i > 0$, $r \in \mathbb{F}$ and any $\mathbf{v} \in \mathbb{F}^{n^{(i-1)}}$, let $\mathbf{u} = \mathsf{Fold}(\mathbf{v}, r)$. Then $\tilde{f}_{\mathbf{u}}(\mathbf{X}) = \tilde{f}_{\mathbf{v}}(r, \mathbf{X})$. Therefore, if we start from $\mathbf{v} \in \mathbb{F}^{n^{(0)}} =$

$\mathbb{F}^{2^\mu}$, after μ rounds of folding with challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$, the resulting message is the constant $\tilde{f}_{\mathbf{v}}(\mathbf{r})$.

We will use the following proposition in this work, which is based on the following observation: the folding applies to every two entries in the vector, so a vector with m non-zero entries will have at most m non-zero entries after folding. By linearity of the code, the number of positions where a vector differs from any codeword should not increase by the folding, so the relative distance should not increase by more than a factor of 2.

Proposition 1. *Folding a vector does not increase the distance to the code by more than a factor of* 2*. Formally, for any* $\mathbf{w} \in \mathbb{F}^{N^{(i)}}$ *and any* $r \in \mathbb{F}$*,*

$$\Delta(\mathsf{Fold}(\mathbf{w}, r), C^{(i+1)}) \leq 2 \cdot \Delta(\mathbf{w}, C^{(i)}).$$

Protocol 2 (FRI). Let $C^{(0)}, \ldots, C^{(\mu)}$ be a foldable family of Reed-Solomon codes with $C^{(i)} = \mathsf{RS}[\mathbb{F}, L^{(i)}, \rho]$ and $|L^{(i)}| = |L^{(i-1)}|/2$. Let $N^{(i)} = |L^{(i)}|$ and $n^{(i)} = 2^{\mu-i} = \rho \cdot N^{(i)}$.

Prover input: Message vector $\mathbf{v} \in \mathbb{F}^{n^{(0)}}$.

Verifier input: Oracle access to codeword $\mathbf{c} \in \mathbb{F}^{N^{(0)}}$.

The protocol proceeds in the following two phases:

Commit phase: For $i = 0, \ldots, \mu - 1$:

1. The verifier samples $r_i \xleftarrow{\$} \mathbb{F}$ and sends it to the prover.
2. The prover computes $\mathbf{c}^{(i+1)} = \mathsf{Fold}(\mathbf{c}^{(i)}, r_i)$ and sends the oracle of $\mathbf{c}^{(i+1)}$ to the verifier (where $\mathbf{c}^{(0)} = \mathbf{c}$). If $i = \mu - 1$, the prover sends the message $\mathbf{v}^{(\mu)}$ (a vector of size $n^{(\mu)} = 1$, i.e., a scalar $v^{(\mu)}$) encoded in $\mathbf{c}^{(\mu)}$ directly.

Query phase: We ignore the details of the query phase. For the purpose of this paper, we point out that the query phase is a linear function of all the codewords sent by the prover during the commit phase. Therefore, we model the query phase as the verifier sampling a (sparse) matrix $\mathbf{Q}$ of size $q \times M$, where $M = \sum_{i=0}^{\mu} N^{(i)}$ and q is an integer that can be derived from the FRI protocol. The verifier accepts if $\mathbf{Q} \cdot (\mathbf{c}^{(0)}, \cdots, \mathbf{c}^{(\mu)}) = \mathbf{0}$. The matrix $\mathbf{Q}$ is sparse in the sense that every row has at most 3 non-zero entries.

We will use the following equivalent variation of soundness definition for FRI. To introduce this variation, we define the following property.

Definition 6 (Goodness). *Let* $C^{(0)}, \ldots, C^{(\mu)}$ *be a foldable family of Reed-Solomon codes with* $C^{(i)} = \mathsf{RS}[\mathbb{F}, L^{(i)}, \rho]$ *and* $|L^{(i)}| = |L^{(i-1)}|/2$*. Let* $N^{(i)} = |L^{(i)}|$ *and* $n^{(i)} = 2^{\mu-i} = \rho \cdot N^{(i)}$*. Let* k *be an integer such that* $k \leq \mu$*. For any sequence* $(\mathbf{c}^{(\mu-k)}, r_{\mu-k}, \mathbf{c}^{(\mu-k+1)}, \ldots, r_{\mu-1}, \mathbf{c}^{(\mu)})$*, where* $r_i \in \mathbb{F}$ *and* $\mathbf{c}^{(i)} \in \mathbb{F}^{N^{(i)}}$*, the goodness of this sequence, denoted by*

$$\mathsf{goodness}(\mathbf{c}^{(\mu-k)}, r_{\mu-k}, \mathbf{c}^{(\mu-k+1)}, \ldots, r_{\mu-1}, r_{\mu-k}, \mathbf{c}^{(\mu)}),$$

is defined as the probability that the FRI query phase accepts this sequence.

One direct observation about this property is that the goodness of any sequence is less than or equal to its suffix, since a sequence passes a query only if all its suffixes pass the query.

Proposition 2. *The goodness of any sequence is less than or equal to its suffix. Formally, for any* $0 < k \leq \mu$*,*

$$\begin{aligned} & \mathsf{goodness}(\mathbf{c}^{(\mu-k)}, r_{\mu-k}, \mathbf{c}^{(\mu-k+1)}, \ldots, r_{\mu-1}, r_{\mu-k}, \mathbf{c}^{(\mu)}) \\ \leq\ & \mathsf{goodness}(\mathbf{c}^{(\mu-k+1)}, r_{\mu-k+1}, \mathbf{c}^{(\mu-k+2)}, \ldots, r_{\mu-1}, \mathbf{c}^{(\mu)}). \end{aligned}$$

For any unbounded deterministic prover $\mathcal{P}$, we model it as its next-message function, i.e., $\mathcal{P}(\mathbf{c}^{(0)}, r_0, \cdots, \mathbf{c}^{(i)}, r_i)$ outputs $\mathbf{c}^{(i+1)}$. Therefore, given any $\mathbf{c}^{(0)}$ and $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$, we can derive the sequence $(\mathbf{c}^{(0)}, r_0, \mathbf{c}^{(1)}, r_1, \ldots, \mathbf{c}^{(\mu-1)}, r_{\mu-1}, \mathbf{c}^{(\mu)})$ by repeatedly applying the next-message function of $\mathcal{P}$. We denote this sequence as $\mathcal{P}(\mathbf{c}^{(0)}, \mathbf{r})$. Soundness of FRI is essentially an upper bound on the goodness of the sequence of $\mathcal{P}(\mathbf{c}^{(0)}, \mathbf{r})$ for any $\mathbf{c}^{(0)}$ and uniformly random $\mathbf{r}$.

Theorem 2 (FRI Soundness). *Let* $C^{(0)}, \ldots, C^{(\mu)}$ *be a foldable family of Reed-Solomon codes with* $C^{(i)} = \mathsf{RS}[\mathbb{F}, L^{(i)}, \rho]$ *and* $|L^{(i)}| = |L^{(i-1)}|/2$*. Let* $N^{(i)} = |L^{(i)}|$ *and* $n^{(i)} = 2^{\mu-i} = \rho \cdot N^{(i)}$*. There exist negligible functions* $\varepsilon_{\mathsf{Commit}}(\lambda)$ *and* $\varepsilon_{\mathsf{Query}}(\lambda)$ *and a constant* $\delta_0 \in (\frac{1-\rho}{3}, 1)$ *such that for any unbounded deterministic prover* $\mathcal{P}$ *and any* $\mathbf{c}^{(0)}$ *with* $\Delta(\mathbf{c}^{(0)}, \mathsf{RS}[\mathbb{F}, L, \rho]) \geq \delta_0$*,*

$$\Pr_{\mathbf{r} \xleftarrow{\$} \mathbb{F}^{\mu}} [\mathsf{goodness}(\mathcal{P}(\mathbf{c}^{(0)}, \mathbf{r})) > \varepsilon_{\mathsf{Query}}(\lambda)] \leq \varepsilon_{\mathsf{Commit}}(\lambda).$$

3 Sum-Check and Repulsive Verifier

The standard soundness definition of sum-check, as presented in Theorem 1, characterizes the protocol's security through the probability of acceptance when the claimed sum is incorrect. While this formulation suffices for standalone analysis, it does not directly capture the intrinsic guarantee of sum-check in a form that enables modular composition with other protocols. This section introduces a functional perspective that characterizes sum-check through the behavior of its verifier function, leading to a property called the *repulsive verifier property*.

To formalize this perspective, consider the sum-check protocol from the verifier's point of view. At the end of the μ rounds of interaction, the verifier holds the random challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$ and outputs a value y that is supposed to equal $f(\mathbf{r})$. Crucially, this output is completely determined by the prover's strategy and the claimed sum c: given a deterministic prover strategy (which can be assumed by fixing the prover's randomness to the choice that maximizes the prover's success probability), the verifier's output defines a function of $\mathbf{r}$. This function, denoted $\mathcal{V}_{\mathcal{P},c} : \mathbb{F}^{\mu} \to \mathbb{F}$, is called the *verifier function* of the protocol.

The verifier function provides a new lens for understanding the security of sum-check. Rather than asking whether the verifier accepts, one can ask: what

functions can a prover make the verifier function equal to? More precisely, for a given prover $\mathcal{P}$ and claimed sum c, how close is $\mathcal{V}_{\mathcal{P},c}$ to any polynomial whose sum differs from c? This shift in perspective reveals a fundamental property of the sum-check protocol that is obscured by the traditional acceptance-based formulation.

Definition 7 (Repulsive Verifier Property). *Let $\mathcal{R} = \{(x, w)\}$ be a relation from $\mathsf{S}_{\mathsf{ins}}$ to $\mathsf{S}_{\mathsf{wit}}$ and μ be an integer. An interactive protocol where the verifier has input x has the (φ, ε)-repulsive verifier property with respect to $\mathcal{R}$, where $\varphi : \mathsf{S}_{\mathsf{ins}} \times \mathsf{S}_{\mathsf{wit}} \times \mathbb{F}^{\mu} \to \mathbb{F}$, if for any (computationally unbounded) prover $\mathcal{P}$ and any $(x, w) \notin \mathcal{R}$:*

$$\Pr_{\mathbf{r} \xleftarrow{\$} \mathbb{F}^{\mu}} \left[\mathcal{V}_{\mathcal{P},x}(\mathbf{r}) = \varphi(x, w, \mathbf{r})\right] \leq \varepsilon.$$

Intuitively, the verifier function is "repelled" from the function $\varphi(x, w, \cdot)$.

For the sum-check protocol, we instantiate this general definition as follows. The relation $\mathcal{R}$ is defined as $\mathcal{R} = \{(c, f) : \sum_{\mathbf{b} \in \{0,1\}^{\mu}} f(\mathbf{b}) = c\}$. The verifier input is the claimed sum c, and the function from which the verifier is repelled is $\varphi(c, f, \mathbf{r}) = f(\mathbf{r})$ whenever c is not the correct sum of f. The repulsive property captures the fact that the verifier function cannot be made to coincide with evaluations of polynomials whose sum is not the claimed value c.

The repulsive property captures a stronger guarantee than standard soundness. While soundness only ensures that a cheating prover cannot make the verifier accept with high probability, repulsiveness ensures that the verifier function itself—regardless of any subsequent checks—cannot be made to coincide with an incorrect polynomial. This distinction is crucial for composition: when sum-check is interleaved with another protocol, the verifier function remains well-defined and its repulsive property continues to hold independently of how the other protocol processes its output.

Theorem 3 (Sum-Check has Repulsive Verifier Property). *The sum-check protocol has the $(\varphi, d\mu/|\mathbb{F}|)$-repulsive verifier property for the relation*

$$\mathcal{R}^{(\mu)} := \{(c, f) : c \in \mathbb{F}, f \in \mathbb{F}^{d}[X_0, \cdots, X_{\mu-1}], \sum_{\mathbf{b} \in \{0,1\}^{\mu}} f(\mathbf{b}) = c\}$$

and the function $\varphi(c, f, \mathbf{r}) = f(\mathbf{r})$.

Proof. We prove by induction on the number of variables μ. For $\mu = 1$, i.e., f is univariate polynomial, the prover directly sends f to the verifier, and the verifier checks $f(0) + f(1) = c$, then samples r_0 and directly outputs $f(r_0)$. In this case, for any $\mathcal{P}$ and any $(c, f) \notin \mathcal{R}^{(1)}$, i.e., $f(0) + f(1) \neq c$ by definition, the prover either sends f, causing the verifier to reject immediately, or sends a polynomial g such that $g(0) + g(1) = c$ and $g \neq f$ as polynomials. Therefore, $\mathcal{V}_{\mathcal{P},c}$ is either the constant function that outputs $\perp$, or is equal to the polynomial g sent from $\mathcal{P}$. In the latter case, by Schwartz-Zippel Lemma, $\mathcal{V}_{\mathcal{P},c}(r_0)$ is equal to $f(r_0)$ with probability at most $d/|\mathbb{F}|$.

Next, for general μ, for any prover $\mathcal{P}$ and any $(c, f) \notin \mathcal{R}^{(\mu)}$, let g be the univariate polynomial sent from $\mathcal{P}$ in the first round. Consider two cases:

– If g is computed correctly from f as specified in the sum-check protocol, then $g(0) + g(1) \neq c$. In this case, $\mathcal{V}_{\mathcal{P},c}$ is the constant $\perp$ function.
– If g is different from the honestly computed polynomial, then by Schwartz-Zippel Lemma, with probability at most $d/|\mathbb{F}|$, $g(r_0) \neq \sum_{\mathbf{b}\in\{0,1\}^{\mu-1}} f(r_0, \mathbf{b})$. Therefore, $(g(r_0), f(r_0, \mathbf{X})) \notin \mathcal{R}^{(\mu-1)}$. By induction hypothesis, $\mathcal{V}_{\mathcal{P},g(r_0)}$ is $(\varphi, d\cdot(\mu-1)/|\mathbb{F}|)$-repulsive with respect to $\mathcal{R}^{(\mu-1)}$, i.e., $\mathcal{V}_{\mathcal{P},g(r_0)}(r_1, \cdots, r_{\mu-1}) = f(r_0, r_1, \cdots, r_{\mu-1})$ with probability $d \cdot (\mu-1)/|\mathbb{F}|$ over $r_1, \cdots, r_{\mu-1}$. By definition, $\mathcal{V}_{\mathcal{P},c}(r_0, r_1, \cdots, r_{\mu-1}) = \mathcal{V}_{\mathcal{P},g(r_0)}(r_1, \cdots, r_{\mu-1})$, which is equal to $f(r_0, r_1, \cdots, r_{\mu-1})$ with probability $d\cdot(\mu-1)/|\mathbb{F}|$ over $r_1, \cdots, r_{\mu-1}$. By union bound, $\mathcal{V}_{\mathcal{P},c}(r_0, r_1, \cdots, r_{\mu-1}) = f(r_0, r_1, \cdots, r_{\mu-1})$ with probability $d \cdot \mu/|\mathbb{F}|$ over $r_0, r_1, \cdots, r_{\mu-1}$, i.e., $\mathcal{V}_{\mathcal{P},c}$ is $(\varphi, d\cdot\mu/|\mathbb{F}|)$-repulsive with respect to $\mathcal{R}^{(\mu)}$. □

The repulsive verifier property provides a clean and self-contained description of what the sum-check protocol enforces, without reference to external mechanisms such as polynomial oracles or commitments. This modular characterization enables the analysis of protocols where sum-check is composed with other components, as shown in subsequent sections.

4 FRI and Attractive Verifier

The FRI protocol (Section 2.7) is conventionally known as a proximity test for Reed–Solomon codes. Yet, it implicitly provides a random polynomial evaluation: once the verifier receives the final constant y, this value is meant to represent the evaluation of a multilinear polynomial $f(\mathbf{X})$ at the randomly chosen challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$. This section formalizes this insight through the *attractive verifier property*, which captures how the verifier's output is constrained to the evaluation of a specific polynomial when the input word is close to a codeword.

4.1 The Attractive Verifier Property

The attractive verifier property captures the constraint that, when the prover's initial oracle is within a small distance of a valid codeword, the verifier's output in the FRI protocol must correspond to the evaluation of the nearest polynomial at the challenge point; otherwise, the verifier rejects. This formalizes the intuition that the verifier's output is attracted to the evaluation of a specific polynomial.

Definition 8 (Attractive Verifier Property). *Let $\mathcal{R}$ be a relation from $\mathsf{S}_{\mathsf{ins}}$ to $\mathsf{S}_{\mathsf{wit}}$ and let μ be a positive integer. An interactive protocol with verifier input x is said to possess the attractive verifier property with parameters (φ, ε), with respect to $\mathcal{R}$, where $\varphi : \mathsf{S}_{\mathsf{ins}} \times \mathsf{S}_{\mathsf{wit}} \times \mathbb{F}^{\mu} \to \mathbb{F}$, if for every (computationally unbounded) prover $\mathcal{P}$ and every $(x, w) \in \mathcal{R}$ the following holds:*

$$\Pr_{\mathbf{r}\xleftarrow{\$}\mathbb{F}^{\mu}} \left[\mathcal{V}_{\mathcal{P},x}(\mathbf{r}) \in \{\varphi(x, w, \mathbf{r}), \perp\}\right] \geq 1 - \varepsilon,$$

where $\perp$ denotes verifier rejection.

In the context of FRI, we instantiate the attractive verifier property as follows. The relation $\mathcal{R}$ consists of all pairs $([\mathbf{w}], \mathbf{w})$ of an oracle commitment and its content. The attracted function φ is given by $\varphi([\mathbf{w}], \mathbf{w}, \mathbf{r}) = f(\mathbf{r})$ if $\mathbf{w}$ is δ_0-close to a unique codeword encoding a multilinear polynomial f, and $\varphi([\mathbf{w}], \mathbf{w}, \mathbf{r}) = \perp$ if $\mathbf{w}$ is δ_0-far from every codeword (thereby guaranteeing rejection).

4.2 FRI Satisfies the Attractive Property

For clarity of exposition and to highlight the essential proof structure, we demonstrate a conservative instantiation of the attractive verifier property for FRI, with the distance bound set to $(1 - \rho)/4$. This choice permits a black-box appeal to FRI's standard soundness guarantee, while the triangle inequality yields a succinct and transparent proof. The extension to tighter distance parameters (for instance, the list-decoding regime) is left for future work.

Theorem 4 (FRI Satisfies the Attractive Verifier Property). *Let $\rho \in (0, 1)$ be the code rate, μ the folding dimension, and ℓ the number of queries. Consider a foldable family of Reed–Solomon codes $C^{(0)}, \ldots, C^{(\mu)}$ with $C^{(i)} = \mathsf{RS}[\mathbb{F}, L^{(i)}, \rho]$ and $|L^{(i)}| = 2^{\mu-i}n$ (where $n = \rho N^{(0)}$). Let $\delta \in (0, \frac{1-\rho}{4})$ be a proximity parameter. Define the function φ_δ on instances $[\mathbf{w}]$ and witness $\mathbf{w}$ by*

$$\varphi_\delta([\mathbf{w}], \mathbf{w}, \mathbf{r}) = \begin{cases} f(\mathbf{r}) & \text{if } \mathbf{w} \text{ is } \delta\text{-close to a (unique) codeword encoding } f, \\ \perp & \text{otherwise.} \end{cases}$$

Then there exist negligible functions $\varepsilon_{\mathsf{Commit}}(\lambda)$ and $\varepsilon_{\mathsf{Query}}(\lambda)$ such that the FRI protocol (with respect to the relation $\mathcal{R} = \{([\mathbf{w}], \mathbf{w}) \mid \mathbf{w} \in \mathbb{F}^{N^{(0)}}\}$) has the $(\varphi_\delta, \varepsilon)$-attractive verifier property, where $\varepsilon = \varepsilon_{\mathsf{Commit}}(\lambda) + \max(\varepsilon_{\mathsf{Query}}(\lambda), (1 - \delta)^\ell)$.

Proof. Fix a deterministic unbounded prover $\mathcal{P}$ and an initial oracle $\mathbf{w}^{(0)} \in \mathbb{F}^{N^{(0)}}$. The interaction defines a sequence $\mathbf{w}^{(0)}, r_0, \mathbf{w}^{(1)}, r_1, \ldots, r_{\mu-1}, \mathbf{w}^{(\mu)}$, where each r_i is sampled uniformly from $\mathbb{F}$ and $\mathbf{w}^{(i)}$ is produced by the prover's next-message function. The final message $\mathbf{w}^{(\mu)}$ is a single scalar. Recall from Section 2.7 that $\mathsf{goodness}(\mathbf{w}^{(0)}, r_0, \mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(\mu)})$ denotes the probability that the verifier accepts in the FRI query phase given this transcript, and that adding a prefix of interaction reduces goodness: for $i \geq 1$, $\mathsf{goodness}(\mathbf{w}^{(i-1)}, r_{i-1}, \ldots, \mathbf{w}^{(\mu)}) \leq \mathsf{goodness}(\mathbf{w}^{(i)}, r_i, \ldots, \mathbf{w}^{(\mu)})$. Also recall that $\mathcal{P}(\mathbf{w}^{(0)}, \mathbf{r})$ is the deterministic mapping from $\mathbf{w}^{(0)}$ and the challenge vector $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$ to the full transcript.

The soundness of FRI guarantees that for any prover, any initial oracle $\mathbf{w}^{(0)}$ that is δ-far from the code, and uniformly random challenges, the probability that $\mathsf{goodness}(\mathbf{w}^{(0)}, r_0, \ldots, \mathbf{w}^{(\mu)}) > \varepsilon_{\mathsf{Query}}(\lambda)$ is at most $\varepsilon_{\mathsf{Commit}}(\lambda)$. We bound the probability of the event

$$E = \left\{\mathbf{w}^{(\mu)} \neq \varphi_\delta([\mathbf{w}^{(0)}], \mathbf{w}^{(0)}, \mathbf{r}) \wedge \mathsf{goodness}(\mathbf{w}^{(0)}, r_0, \ldots, \mathbf{w}^{(\mu)}) > \max(\varepsilon_{\mathsf{Query}}(\lambda), (1-\delta)^\ell)\right\},$$

where $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$. We distinguish two cases based on the distance of $\mathbf{w}^{(0)}$ to the code $C^{(0)}$.

Case 1: $\mathbf{w}^{(0)}$ is δ-far from every codeword. In this case $\varphi_\delta([\mathbf{w}^{(0)}], \mathbf{w}^{(0)}, \mathbf{r}) = \perp$. By FRI soundness, the probability over random $\mathbf{r}$ that $\mathsf{goodness} > \varepsilon_{\mathsf{Query}}(\lambda)$ is at most $\varepsilon_{\mathsf{Commit}}(\lambda)$. Consequently, the verifier outputs a value different from $\perp$ with probability at most $\varepsilon_{\mathsf{Commit}}(\lambda) + \varepsilon_{\mathsf{Query}}(\lambda)$.

Case 2: $\mathbf{w}^{(0)}$ is δ-close to some codeword. Because $\delta < (1-\rho)/4 < (1-\rho)/2$, the nearest codeword is unique; write it as $\mathsf{Enc}(f^{(0)})$ for a multilinear polynomial $f^{(0)}$. Hence $\varphi_\delta([\mathbf{w}^{(0)}], \mathbf{w}^{(0)}, \mathbf{r}) = f^{(0)}(\mathbf{r})$.

Define $\mathcal{A}$ as the set of challenge vectors $\mathbf{r}$ for which every oracle $\mathbf{w}^{(i)}$ in the transcript $\mathcal{P}(\mathbf{w}^{(0)}, \mathbf{r})$ is δ-close to the corresponding code $C^{(i)}$. We analyze the conditional behavior on $\mathbf{r} \in \mathcal{A}$ and $\mathbf{r} \notin \mathcal{A}$.

Subcase 2a: $\mathbf{r} \in \mathcal{A}$. For each $i = 0, \ldots, \mu$, let $f^{(i)}$ be the unique polynomial such that $\mathbf{w}^{(i)}$ is δ-close to $\mathsf{Enc}(f^{(i)})$. Since $\mathbf{w}^{(\mu)}$ is a constant, $f^{(\mu)} = \mathbf{w}^{(\mu)}$. If $f^{(\mu)} \neq f^{(0)}(r_0, \ldots, r_{\mu-1})$, then there is a smallest index $i \in \{1, \ldots, \mu\}$ with $f^{(i)}(\mathbf{X}) \neq f^{(i-1)}(r_{i-1}, \mathbf{X})$. For that i, distinct codewords of $C^{(i)}$ have relative distance at least $1 - \rho > 4\delta$, so $\Delta(\mathsf{Enc}(f^{(i-1)}(r_{i-1}, X)), \mathsf{Enc}(f^{(i)})) > 4\delta$. Folding does not increase distance by more than a factor of 2, hence

$$\Delta(\mathsf{Fold}(\mathbf{w}^{(i-1)}, r_{i-1}), \mathsf{Enc}(f^{(i-1)}(r_{i-1}, X))) < 2\delta.$$

Applying the triangle inequality,

$$\begin{aligned}\Delta(\mathsf{Fold}(\mathbf{w}^{(i-1)}, r_{i-1}), \mathbf{w}^{(i)}) \geq\ & \Delta(\mathsf{Enc}(f^{(i-1)}(r_{i-1}, X)), \mathsf{Enc}(f^{(i)})) \\ & - \Delta(\mathsf{Fold}(\mathbf{w}^{(i-1)}, r_{i-1}), \mathsf{Enc}(f^{(i-1)}(r_{i-1}, X))) \\ & - \Delta(\mathbf{w}^{(i)}, \mathsf{Enc}(f^{(i)})) \\ >\ & 4\delta - 2\delta - \delta = \delta.\end{aligned}$$

Thus $\mathbf{w}^{(i)}$ differs from $\mathsf{Fold}(\mathbf{w}^{(i-1)}, r_{i-1})$ on more than a δ fraction of coordinates. By the construction of the FRI query phase, this guarantees $\mathsf{goodness} \leq (1-\delta)^\ell$. Therefore, the event E cannot occur when $\mathbf{r} \in \mathcal{A}$ and $\mathbf{w}^{(\mu)} \neq f^{(0)}(\mathbf{r})$.

Subcase 2b: $\mathbf{r} \notin \mathcal{A}$. Partition the challenge space as $\mathbb{F}^\mu = \mathcal{A} \cup \left(\bigcup_{i=1}^{\mu} \mathcal{B}_i\right)$, where $\mathcal{B}_i$ consists of those $\mathbf{r}$ for which i is the smallest index with $\Delta(\mathbf{w}^{(i)}, C^{(i)}) > \delta$. Fix i and a prefix $(r_0, \ldots, r_{i-1})$ that determines $\mathbf{w}^{(i)}$ being δ-far from $C^{(i)}$. By FRI soundness applied to this remainder subprotocol, the probability over $r_i, \ldots, r_{\mu-1}$ that

$$\mathsf{goodness}(\mathbf{w}^{(i)}, r_i, \ldots, \mathbf{w}^{(\mu)}) > \varepsilon_{\mathsf{Query}}(\lambda)$$

is at most $\varepsilon_{\mathsf{Commit}}(\lambda)$. Since adding a prefix reduces goodness, the same bound holds for $\mathsf{goodness}(\mathbf{w}^{(0)}, r_0, \ldots, \mathbf{w}^{(\mu)})$. Averaging over all prefixes defining $\mathcal{B}_i$, we obtain

$$\Pr_{\mathbf{r} \in \mathcal{B}_i}\left[\mathsf{goodness}(\mathbf{w}^{(0)}, r_0, \ldots, \mathbf{w}^{(\mu)}) > \varepsilon_{\mathsf{Query}}(\lambda)\right] \leq \varepsilon_{\mathsf{Commit}}(\lambda).$$

Taking the union over $i = 1, \ldots, \mu$, the probability over $\mathbf{r} \notin \mathcal{A}$ that $\mathsf{goodness}$ exceeds $\varepsilon_{\mathsf{Query}}(\lambda)$ is at most $\varepsilon_{\mathsf{Commit}}(\lambda)$.

Conclusion. In Case 1, any acceptance (i.e., output $\neq \perp$) violates the attractive property; this occurs with probability at most $\varepsilon_{\mathsf{Commit}}(\lambda) + \varepsilon_{\mathsf{Query}}(\lambda)$. In Case 2, the combination of Subcases 2a and 2b shows that the probability of the event E is bounded by the probability that $\mathbf{r} \notin \mathcal{A}$ and $\mathsf{goodness} > \varepsilon_{\mathsf{Query}}(\lambda)$, which is at most $\varepsilon_{\mathsf{Commit}}(\lambda)$. Hence, overall, the verifier outputs a value $\mathbf{w}^{(\mu)} \notin \{\varphi_\delta([\mathbf{w}^{(0)}], \mathbf{w}^{(0)}, \mathbf{r}), \perp\}$ with probability at most $\varepsilon_{\mathsf{Commit}}(\lambda) + \max(\varepsilon_{\mathsf{Query}}(\lambda), (1-\delta)^\ell)$. This matches the claimed ε-attractive verifier property. □

4.3 Application: Modular Analysis of BaseFold

With both the repulsive and attractive verifier properties established, we can now provide a modular analysis of the BaseFold polynomial commitment scheme. BaseFold combines sum-check and FRI in a parallel execution where both protocols share the same randomness $\mathbf{r}$.

Protocol 3 (BaseFold PCS). Let $C^{(0)}, \ldots, C^{(\mu)}$ be a foldable code family as defined in Section 2.7, with $C^{(i)} = \mathsf{RS}[\mathbb{F}, L^{(i)}, \rho]$, $n^{(i)} = 2^{\mu-i}$, and rate $\rho \in (0,1)$.

- $\mathsf{Setup}(1^\lambda, 2^\mu) \to \mathsf{pp}$: Choose a cryptographic hash function. Output public parameters pp including the code family description and the hash function. The verifier parameters vp equal pp.
- $\mathsf{Commit}(\mathsf{pp}, f) \to \mathsf{cm}$: Let $\mathbf{f}$ be the coefficients of the μ-variate multilinear polynomial $f(\mathbf{X})$. Compute the encoding $\mathbf{w} = \mathsf{Encode}_{C^{(0)}}(\mathbf{f})$. Output the string oracle to $\mathbf{w}$ as the commitment $\mathsf{cm} = [\mathbf{w}]$. (In practice, the prover retains $\mathbf{w}$.)
- $\mathsf{Open}(\mathsf{pp}, \mathsf{cm}, f) \to b$: Compute $\mathbf{w}' = \mathsf{Encode}_{C^{(0)}}(\mathbf{f})$. Output 1 if $\Delta(\mathbf{w}', \mathbf{w}) < (1-\rho)/4$, and 0 otherwise.
- $\mathsf{Eval}(\mathsf{vp}, \mathsf{cm}, \mathbf{z}, y; f, \mathsf{pp}) \to b$:
  1. **Parallel execution:** Execute sum-check and FRI in parallel, sharing the same challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$:
     - **Sum-check:** Prover input: f, $\mathbf{z}$, y. Verifier input: y, $\mathbf{z}$. For polynomial $f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$ with claimed sum y. In each round $i \in \{0, \ldots, \mu-1\}$, prover sends $g_i(X_i)$, verifier checks and samples r_i.
     - **FRI:** Prover input: $\mathbf{w}$. Verifier input: oracle access to $\mathsf{cm} = [\mathbf{w}]$. In each round $i \in \{0, \ldots, \mu-1\}$, prover sends oracle $\mathbf{w}^{(i)}$, verifier samples r_i (sharing the same r_i as in sum-check).
  2. **Final check:** Let y_{sc} be sum-check's final value and y_{fri} be FRI's output. Verifier accepts if:
     - FRI does not reject (i.e., $y_{\mathsf{fri}} \neq \perp$)
     - $y_{\mathsf{sc}} = y_{\mathsf{fri}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$

The security of BaseFold follows from a clean composition of the two functional properties.

Theorem 5 (BaseFold Security). *Fix $\rho \in (0,1)$ and $\delta \in (0, \frac{1-\rho}{4})$, and define*

$$\varepsilon_{\mathsf{BF}}(\lambda) := \frac{2\mu}{|\mathbb{F}|} + \varepsilon_{\mathsf{Commit}}(\lambda) + \max(\varepsilon_{\mathsf{Query}}(\lambda), (1-\delta)^\ell).$$

The BaseFold construction satisfies the security properties of Definition 3 with perfect correctness, perfect completeness, perfect binding, and knowledge soundness for exact commitments with knowledge error at most $\varepsilon_{\mathsf{BF}}(\lambda)$.

Proof. **Correctness.** If $\mathsf{cm} = [\mathbf{w}]$ is generated by $\mathsf{Commit}(\mathsf{pp}, f)$, then by definition $\mathbf{w} = \mathsf{Encode}_C(\mathbf{f})$. Hence $\mathsf{Open}(\mathsf{pp}, \mathsf{cm}, f)$ recomputes the same encoding and accepts.

Completeness. Fix any multilinear polynomial f and any point $\mathbf{z} \in \mathbb{F}^\mu$, and let $\mathsf{cm} = [\mathbf{w}]$ with $\mathbf{w} = \mathsf{Encode}_C(\mathbf{f})$. In the sum-check component, the prover and verifier run sum-check on the polynomial

$$h(\mathbf{X}) := f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$$

with claimed sum $f(\mathbf{z})$. By the identity $\sum_{\mathbf{b}\in\{0,1\}^\mu} f(\mathbf{b}) \cdot \mathsf{eq}(\mathbf{b}, \mathbf{z}) = f(\mathbf{z})$, honest sum-check accepts and outputs

$$y_{\mathsf{sc}} = h(\mathbf{r}) = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}).$$

In the FRI component, since $\mathbf{w}$ is an honestly encoded codeword, honest FRI accepts and outputs

$$y_{\mathsf{fri}} = f(\mathbf{r}).$$

Therefore the final check

$$y_{\mathsf{sc}} = y_{\mathsf{fri}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$$

holds, so Eval accepts with probability 1.

Binding. Suppose $\mathsf{Open}(\mathsf{pp}, [\mathbf{w}], f_0) = \mathsf{Open}(\mathsf{pp}, [\mathbf{w}], f_1) = 1$. Then

$$\mathsf{Encode}_C(\mathbf{f}_0) = \mathbf{w} = \mathsf{Encode}_C(\mathbf{f}_1).$$

Since the encoding map is injective, we obtain $\mathbf{f}_0 = \mathbf{f}_1$, and hence $f_0 = f_1$. So the scheme is perfectly binding.

Knowledge soundness. Define the extractor $\mathcal{E}$ as follows: on input $(\mathsf{pp}, [\mathbf{w}], \mathbf{z}, y)$, read the oracle $[\mathbf{w}]$, recover the committed codeword $\mathbf{w}$, and output the unique multilinear polynomial f with coefficient vector $\mathbf{f}$ such that $\Delta(\mathbf{w}, \mathsf{Encode}_C(\mathbf{f})) < (1-\rho)/4$. If no such polynomial exists, then output $\perp$. This extractor is deterministic and runs in polynomial time; it does not need to use rewinding, although it may be viewed as having oracle access to the prover's next-message function. We show that the probability of verifier acceptance (i.e., $b = 1$) and $((\mathsf{pp}, [\mathbf{w}], \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}}$ is at most $\varepsilon_{\mathsf{BF}}(\lambda)$.

Let y_{sc} and y_{fri} be the outputs of the sum-check and FRI components respectively. By the attractive property of FRI (Theorem 4), f is simply the function $\varphi_\delta([\mathbf{w}], \mathbf{w}, \mathbf{r})$, and for any prover $\mathcal{P}^*$ and any instance $(\mathsf{pp}, [\mathbf{w}], \mathbf{z}, y)$,

$$\Pr_{\mathbf{r}\xleftarrow{\$}\mathbb{F}^\mu}\left[y_{\mathsf{fri}} \notin \{f(\mathbf{r}), \perp\}\right] \le \varepsilon_{\mathsf{Commit}}(\lambda) + \max(\varepsilon_{\mathsf{Query}}(\lambda), (1-\delta)^\ell).$$

On the complementary event where $y_{\mathsf{fri}} \in \{f(\mathbf{r}), \perp\}$, acceptance requires $y_{\mathsf{fri}} \neq \perp$, hence necessarily $y_{\mathsf{fri}} = f(\mathbf{r})$ and $f \neq \perp$. The assumption $((\mathsf{pp}, [\mathbf{w}], \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}}$ implies $f(\mathbf{z}) \neq y$. The final equality test then forces

$$y_{\mathsf{sc}} = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}).$$

Now sum-check is being applied to the polynomial $h(\mathbf{X}) = f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$, which has degree at most 2 in each variable, and whose claimed sum is $y \neq f(\mathbf{z})$. By Theorem 3,

$$\Pr\big[y_{\mathsf{sc}} = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})\big] = \Pr\big[y_{\mathsf{sc}} = h(\mathbf{r})\big] \leq \frac{2\mu}{|\mathbb{F}|}.$$

Combining the two bounds with the union bound gives

$$\Pr\big[b = 1 \wedge ((\mathsf{pp}, [\mathbf{w}], \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}}\big] \leq \frac{2\mu}{|\mathbb{F}|} + \varepsilon_{\mathsf{Commit}}(\lambda) + \max(\varepsilon_{\mathsf{Query}}(\lambda), (1-\delta)^{\ell}) = \varepsilon_{\mathsf{BF}}(\lambda).$$

□

This modular analysis demonstrates the power of the functional perspective: the security of BaseFold reduces to the simple observation that the verifier function cannot simultaneously be repelled from an incorrect polynomial and attracted to a fixed polynomial unless the fixed polynomial is correct. The analysis is self-contained and does not require reasoning about the intricate interaction between the two protocols at the level of individual messages.

5 BulletProofs and BulletEval

This section applies the same philosophy to BulletProofs, a widely deployed polynomial commitment scheme based on discrete logarithm assumptions. The key difference is that BulletProofs operates in a computational setting, where the attractive property must be formulated in terms of knowledge extraction rather than information-theoretic bounds.

BulletProofs, introduced by Bünz et al. [11], is an inner-product argument that can be used as a polynomial commitment scheme. In the polynomial commitment setting, the prover commits to a multilinear polynomial $f(\mathbf{X})$ through a Pedersen commitment $C = \langle \mathbf{f}, \mathbf{G} \rangle$, where $\mathbf{f} = f(\{0,1\}^{\mu})$, $\mathbf{G}$ is a vector of group generators. The prover later proves an evaluation claim $f(\mathbf{z}) = y$.

A key observation by Bootle et al. [10] is that BulletProofs can be understood as a sum-check-style protocol. The core mechanism involves repeated folding: in each round, the prover splits the current vector into two halves, sends cross-term commitments, and the verifier provides a random challenge to combine them. After μ rounds, the protocol reduces to checking a simple group equation.

From a functional perspective, BulletProofs can be decomposed into two parallel components that share the same randomness:

1. A scalar-valued sum-check for the polynomial $f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$ with claimed sum y. This component provides the repulsive property that rejects incorrect evaluation claims.
2. A group-valued sum-check for the polynomial $f(\mathbf{X}) \cdot \tilde{\mathbf{G}}(\mathbf{X})$, where $\tilde{\mathbf{G}}(\mathbf{X})$ is the group-valued multilinear extension of the generators $\mathbf{G}$, with claimed sum C. This component binds the evaluation to the committed polynomial.

The first component is the same as in BaseFold, while the second component provides an alternative to FRI for attracting the verifier output to a specific polynomial. We isolate the second component as a standalone protocol called BulletEval, which provides the computational analogue of the attractive property.

5.1 The BulletEval Protocol

BulletEval is an interactive protocol derived from the group-valued sum-check component of BulletProofs. It provides polynomial random evaluation functionality in a computational setting.

Protocol 4 (BulletEval). Let $\mathbf{G} = (G_0, \ldots, G_{n-1}) \in \mathbb{G}^n$ be uniformly sampled, where $n = 2^\mu$, and let $\tilde{\mathbf{G}}(\mathbf{X})$ be the group-valued multilinear polynomial interpolated from $\mathbf{G}$. Let $f(\mathbf{X})$ be a scalar-valued multilinear polynomial with coefficients $\mathbf{f} \in \mathbb{F}^n$, with commitment $C = \langle \mathbf{f}, \mathbf{G} \rangle$.

Prover input: Vector $\mathbf{f}$, randomness r, and the verifier's inputs.
Verifier input: Commitment C, generators $\mathbf{G}$.
Output: The verifier outputs $s = f(\mathbf{r})$ or $\perp$ where $\mathbf{r}$ is the random challenge vector sampled during the protocol.

1. Initialize $C^{(0)} = C$. For $i = 0, \ldots, \mu - 1$:
   (a) The prover computes $H_i(X) = \sum_{\mathbf{b} \in \{0,1\}^{\mu-i-1}} f(r_0, \ldots, r_{i-1}, X, \mathbf{b}) \cdot \tilde{\mathbf{G}}(r_0, \ldots, r_{i-1}, X, \mathbf{b})$ and sends $H_i(X)$ to the verifier.
   (b) The verifier checks that $\deg(H_i(X)) \leq 2$ and $H_i(0) + H_i(1) = C^{(i)}$.
   (c) The verifier sends a random challenge $r_i \in \mathbb{F}$.
   (d) Both parties compute $C^{(i+1)} = H_i(r_i)$.
2. After μ rounds, both parties obtain $C^{(\mu)}$.
3. The prover computes $s = f(\mathbf{r})$ and sends it to the verifier.
4. The verifier computes $\tilde{\mathbf{G}}(\mathbf{r})$ from the folded generators and checks $C^{(\mu)} = s \cdot \tilde{\mathbf{G}}(\mathbf{r})$. If the check passes, output s; otherwise output $\perp$.

Equivalently, the protocol can be viewed as a group-valued sum-check: the prover is proving that the sum of $f(\mathbf{X}) \cdot \tilde{\mathbf{G}}(\mathbf{X})$ over the hypercube equals C, and the final output is the evaluation at the random challenge point. However, the goal of this protocol is reversed: instead of proving the validity of C by verifying $f(\mathbf{r}) \cdot \tilde{\mathbf{G}}(\mathbf{r})$, the protocol exploits the binding property of the Pedersen commitment, relying on the fact that C is the correct sum to establish the correctness of $f(\mathbf{r}) \cdot \tilde{\mathbf{G}}(\mathbf{r})$, and then proving that the output equals $f(\mathbf{r})$.

The key difference between BulletEval and the information-theoretic attractive property of FRI is that BulletEval operates in a group setting where the prover's messages are commitments rather than field elements. This introduces computational assumptions into the security analysis.

5.2 Knowledge Attractive Property

In the computational setting, the attractive property must be formulated in terms of knowledge extraction: if a prover can make the verifier output a value, then an extractor can recover the underlying polynomial.

Definition 9 (Knowledge Attractive Property). *A protocol with verifier input x has the (φ, ε)-knowledge attractive property, where $\varphi : \mathsf{S}_{\mathsf{pp}} \times \mathsf{S}_{\mathsf{ins}} \times \mathsf{S}_{\mathsf{wit}} \times \mathbb{F}^{\mu} \to \mathbb{F}$, if there exists an expected polynomial-time extractor $\mathcal{E}$ such that for any PPT prover strategies $\mathcal{P}_1, \mathcal{P}_2$:*

$$\Pr_{\substack{\mathbf{r} \xleftarrow{\$} \mathbb{F}^{\mu} \\ \mathsf{pp} \xleftarrow{\$} \mathsf{S}_{\mathsf{pp}}}} \left[y \notin \{\varphi(\mathsf{pp}, x, w, \mathbf{r}), \perp\} \;\middle|\; \begin{array}{l} \mathcal{P}_1(\mathsf{pp}) \to x, \mathsf{st} \\ \langle \mathcal{P}_2(\mathsf{pp}, x, \mathsf{st}), \mathcal{V}(\mathsf{pp}, x) \rangle \to y \\ \mathcal{E}^{\mathsf{nextmsg}[\mathcal{P}_2]}(\mathsf{pp}, x, \mathsf{st}) \to w \end{array} \right] \leq \varepsilon,$$

where $\mathbf{r}$ is the random challenge that the verifier samples during the interaction, and the probability also includes the randomness of the prover and the extractor.

This definition no longer involves the relation $\mathcal{R}$, because we removed the quantifier "for all w" and replaced it with the extracted w, there is no need to constrain w with $\mathcal{R}$. Instead, all the requirements on the relation between x and w can be encoded in the function φ. Note that the probability includes the randomness of the public parameter, because for any fixed public parameter, there will always be adversaries that hardcode values to solve the underlying hard problem, e.g., discrete logarithms. Furthermore, we allow the prover to choose the instance dynamically according to the public parameter. The attractive property then requires that the verifier always evaluates φ with the extracted w except with non-negligible probability.

We also allow the verifier output to be $\perp$ to accommodate the situation where the prover has the correct witness that can be extracted by $\mathcal{E}$, but the prover aborts randomly with non-negligible probability (say $1/2$). In that case, $\varphi(\mathsf{pp}, x, w, \mathbf{r})$ would be the normal function but the verifier output y is $\perp$ with probability $1/2$, causing $y \neq \varphi(\mathsf{pp}, x, w, \mathbf{r})$ with high probability. We allow y to be $\perp$ to accommodate these intentionally failing provers.

5.3 BulletEval Satisfies the Knowledge Attractive Property

Theorem 6 (BulletEval is Knowledge Attractive). *Under the q-discrete logarithm assumption in the group $\mathbb{G}$ for $q \geq n$,* BulletEval *has $(\varphi, \mathit{negl}(\lambda))$-knowledge attractive property for the function*

$$\varphi(\mathbf{G}, C, \mathbf{f}, \mathbf{r}) = \begin{cases} \tilde{\mathbf{f}}(\mathbf{r}), & \text{if } C = \langle \mathbf{f}, \mathbf{G} \rangle \\ \perp, & \text{otherwise} \end{cases}$$

where λ is the security parameter.

Proof. This proof is an adaptation of the proof in [15] to our definition of knowledge attractive property. We first note that it suffices to consider deterministic prover strategies, as any randomized prover can be derandomized by fixing its random coins to the value that maximizes the success probability. Since $\mathcal{P}_1$ and $\mathcal{P}_2$ are PPT, this does not affect the asymptotic running time.

Let $\mathcal{P}_1, \mathcal{P}_2$ be arbitrary deterministic PPT strategies. Let $\varepsilon = \Pr_{\mathbf{r},\mathsf{pp}}[y \neq \perp]$ be the probability that the verifier does not abort. If $\varepsilon \leq \mathrm{negl}(\lambda)$, then $\Pr[y \notin \{\varphi(\mathsf{pp}, x, w, \mathbf{r}), \perp\}] \leq \Pr[y \neq \perp] \leq \mathrm{negl}(\lambda)$, and the statement holds trivially for any extractor. We therefore assume ε is non-negligible.

Extractor construction. The extractor $\mathcal{E}$, given oracle access to $\mathsf{nextmsg}[\mathcal{P}_2]$ and input $(\mathsf{pp} = \mathbf{G}, x = C, \mathsf{st})$, constructs a transcript tree as follows.

Root level. Fix $\mathsf{pp} = \mathbf{G}$ and $x = C$ chosen by $\mathcal{P}_1$. The extractor queries $\mathsf{nextmsg}[\mathcal{P}_2]$ with the empty transcript prefix to obtain the first prover message $H_0(X)$, which is a group-valued polynomial. The verifier checks that $\deg(H_0) \leq 2$ and $H_0(0) + H_0(1) = C$.

Building the tree. The extractor builds a tree of depth μ where each non-leaf node has 4 children. For a node at depth i with prefix $r_0, \ldots, r_{i-1}$, the extractor:

1. Samples 4 distinct random challenges $r_i^1, r_i^2, r_i^3, r_i^4 \in \mathbb{F}$.
2. For each challenge r_i^j, continues the transcript by querying $\mathsf{nextmsg}[\mathcal{P}_2]$ with the extended prefix to obtain $H_{i+1}(X)$.
3. At depth $\mu - 1$, after obtaining $H_{\mu-1}(X)$ and evaluating it at the challenge, obtains the final prover message s for each leaf.

If at any point the prover fails to respond properly (e.g., H_i has degree > 2, or the sum-check consistency $H_i(0) + H_i(1) = H_{i-1}(r_{i-1})$ fails), that branch terminates with $\perp$. Since ε is non-negligible, the extractor can obtain 4 distinct accepting branches at each depth by rewinding in expected polynomial time ($O(1/\epsilon)$) via the standard forking lemma.

Consistency conditions. For an accepting transcript, the following conditions hold:

- For each round i: $\deg(H_i) \leq 2$ and $H_i(0) + H_i(1) = C^{(i)}$, where $C^{(0)} = C$ and $C^{(i+1)} = H_i(r_i)$.
- For the final round: $C^{(\mu)} = s \cdot \tilde{\mathbf{G}}(\mathbf{r})$ where $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$.

Equivalently, for each leaf with path $\mathbf{r}$:

$$H_{\mu-1}(r_{\mu-1}) = s \cdot \tilde{\mathbf{G}}(\mathbf{r})$$

and for each i, $H_{i-1}(r_{i-1}) = H_i(0) + H_i(1)$.

Leaf-level polynomial extraction. Consider a node at depth $\mu-1$ with transcript prefix $r_0, \ldots, r_{\mu-2}$. The extractor has 4 challenges $r^1_{\mu-1}, \ldots, r^4_{\mu-1}$ and corresponding leaf values $s_1, \ldots, s_4$ from accepting transcripts.

For each j, we have:

$$C^{(\mu,j)} = H_{\mu-1}(r^j_{\mu-1}) = s_j \cdot \tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, r^j_{\mu-1}).$$

Since $\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, X) = (1-X)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0) + X\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1)$, this becomes:

$$H_{\mu-1}(r^j_{\mu-1}) = s_j \cdot ((1-r^j_{\mu-1})\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0) + r^j_{\mu-1}\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1)).$$

Let $f(X)$ be the unique degree-3 polynomial that interpolates the points $(r^j_{\mu-1}, s_j)$ for $j = 1, \ldots, 4$. Define two auxiliary polynomials:

- $g(X)$ of degree ≤ 2 that interpolates $(r^j_{\mu-1}, (1-r^j_{\mu-1}) \cdot f(r^j_{\mu-1}))$ for $j = 1, 2, 3$,
- $g'(X)$ of degree ≤ 2 that interpolates $(r^j_{\mu-1}, r^j_{\mu-1} \cdot f(r^j_{\mu-1}))$ for $j = 1, 2, 3$.

Then for $j = 1, 2, 3$:

$$H_{\mu-1}(r^j_{\mu-1}) = g(r^j_{\mu-1}) \cdot \tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0) + g'(r^j_{\mu-1}) \cdot \tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1).$$

The left-hand side $H_{\mu-1}(X)$ is a degree-2 group-valued polynomial, and the right-hand side $g(X)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0) + g'(X)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1)$ is also degree ≤ 2. Since they agree on 3 distinct points, they must be identical as polynomials:

$$H_{\mu-1}(X) = g(X)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0) + g'(X)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1).$$

Evaluating this identity at $r^4_{\mu-1}$ and comparing with the leaf condition, we obtain:

$$\begin{aligned}(g(r^4_{\mu-1}) - (1-r^4_{\mu-1})f(r^4_{\mu-1})) \cdot \tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0) \\ + (g'(r^4_{\mu-1}) - r^4_{\mu-1}f(r^4_{\mu-1})) \cdot \tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1) = O.\end{aligned}$$

Here $\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0)$ and $\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1)$ are random independent group elements (since the r_i's are uniformly random and $\mathbf{G}$ is uniformly sampled). By the q-DLOG assumption for $q \geq n$, any non-trivial linear relation among them occurs with only negligible probability. Therefore, with probability $1 - \mathrm{negl}(\lambda)$:

$$g(r^4_{\mu-1}) = (1-r^4_{\mu-1})f(r^4_{\mu-1}) \quad \text{and} \quad g'(r^4_{\mu-1}) = r^4_{\mu-1}f(r^4_{\mu-1}).$$

These equalities now hold for all four points $j = 1, \ldots, 4$. Summing the two equalities gives:

$$g(r^j_{\mu-1}) + g'(r^j_{\mu-1}) = f(r^j_{\mu-1}) \quad \text{for all } j \in \{1, 2, 3, 4\}.$$

The polynomial $g(X) + g'(X)$ has degree ≤ 2, while $f(X)$ has degree ≤ 3. Agreement on 4 distinct points forces these polynomials to be identical, hence $\deg(f) \leq 2$.

Now consider $g(X) = (1 - X)f(X)$ on the same 4 points. g has degree ≤ 2, and with $\deg(f) \leq 2$, the product $(1 - X)f(X)$ has degree ≤ 3. Two degree-3 polynomials agreeing on 4 points are identical, so $g(X) = (1 - X)f(X)$ as polynomials. This forces $\deg(g) = \deg(f) + 1$ unless cancellation occurs; since $\deg(g) \leq 2$, we must have $\deg(f) \leq 1$. Thus $f(X)$ is a linear (multilinear) polynomial.

Consequently, $s_j = f(r_{\mu-1}^j)$ where f is a degree-1 polynomial, and we have:

$$H_{\mu-1}(X) = (1 - X)f(X)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0) + Xf(X)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1).$$

From the sum-check consistency $C^{(\mu-1)} = H_{\mu-1}(0) + H_{\mu-1}(1)$:

$$C^{(\mu-1)} = f(0)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 0) + f(1)\tilde{\mathbf{G}}(r_0, \ldots, r_{\mu-2}, 1).$$

Induction up the tree. We generalize the above by induction. Define the following claim for k from $\mu - 1$ down to 0.

Claim. For any node at depth k (with $0 \leq k \leq \mu - 1$) having challenge prefix $\mathbf{r} = (r_0, \ldots, r_{k-1})$, there exists a $(\mu - k)$-variate multilinear polynomial $f_{\mathbf{r}}(\mathbf{X})$ such that:

$$C^{(k)} = \sum_{\mathbf{b} \in \{0,1\}^{\mu-k}} f_{\mathbf{r}}(\mathbf{b}) \cdot \tilde{\mathbf{G}}(\mathbf{r}, \mathbf{b})$$

and for every leaf path extending this prefix with challenges $r_k, \ldots, r_{\mu-1}$, the leaf value s satisfies $s = f_{\mathbf{r}}(r_k, \ldots, r_{\mu-1})$.

Base case ($k = \mu - 1$). Already proven above: $f_{\mathbf{r}}$ is a 1-variate multilinear polynomial satisfying the claim.

Inductive step. Assume the claim holds for all nodes at depth $k + 1$. Let $\mathbf{r} = (r_0, \ldots, r_{k-1})$ be a prefix of length k. The extractor obtains 4 children at challenges $r_k^1, \ldots, r_k^4$, each giving a $(\mu - k - 1)$-variate multilinear polynomial $f_{\mathbf{r},r_k^j}$ by the induction hypothesis.

Let $F(X, \mathbf{Y})$ be the unique polynomial of individual degree ≤ 3 in X and multilinear in $\mathbf{Y}$ that interpolates the 4 points $(r_k^j, f_{\mathbf{r},r_k^j}(\mathbf{Y}))$. From the sum-check consistency at depth k: $C^{(k)} = H_k(0) + H_k(1)$.

For each challenge r_k^j, the induction hypothesis at depth $k + 1$ gives:

$$H_k(r_k^j) = \sum_{\mathbf{b} \in \{0,1\}^{\mu-k-1}} F(r_k^j, \mathbf{b}) \cdot \tilde{\mathbf{G}}(\mathbf{r}, r_k^j, \mathbf{b}).$$

We now apply the same polynomial interpolation argument as in the base case, treating $F(X, \mathbf{b})$ as a degree-3 scalar polynomial in X for each fixed $\mathbf{b}$. Define auxiliary polynomials $G(X, \mathbf{b})$ and $G'(X, \mathbf{b})$ of degree ≤ 2 in X by interpolating the values at r_k^1, r_k^2, r_k^3. By the same group-valued polynomial identity argument, the DLOG assumption forces:

$$G(X, \mathbf{b}) = (1 - X)F(X, \mathbf{b}) \quad \text{and} \quad G'(X, \mathbf{b}) = XF(X, \mathbf{b})$$

as polynomials in X, for all $\mathbf{b} \in \{0, 1\}^{\mu-k-1}$ simultaneously.

As in the base case, the equality $G(X, \mathbf{b}) + G'(X, \mathbf{b}) = F(X, \mathbf{b})$ on 4 points forces $F(X, \mathbf{b})$ to have degree ≤ 2 in X. Then $G(X, \mathbf{b}) = (1-X)F(X, \mathbf{b})$ as polynomials forces $F(X, \mathbf{b})$ to have degree ≤ 1 in X. Hence $F(X, \mathbf{Y})$ is multilinear in all $\mu - k$ variables.

The sum-check consistency then yields:

$$C^{(k)} = \sum_{\mathbf{b} \in \{0,1\}^{\mu-k}} F(b_0, b_1, \ldots, b_{\mu-k-1}) \cdot \tilde{\mathbf{G}}(\mathbf{r}, \mathbf{b}),$$

which satisfies the claim at depth k with $f_{\mathbf{r}} = F$.

Root extraction and conclusion. Applying the claim at the root ($k = 0$, $\mathbf{r} = \emptyset$), we obtain a μ-variate multilinear polynomial F such that:

$$C = \sum_{\mathbf{b} \in \{0,1\}^{\mu}} F(\mathbf{b}) \cdot \tilde{\mathbf{G}}(\mathbf{b}) = \langle \mathbf{f}, \mathbf{G} \rangle,$$

where $\mathbf{f} = F(\{0,1\}^{\mu})$. The extractor outputs $w = \mathbf{f}$.

We now prove the aforementioned extractor satisfies the knowledge attractive property. The claim guarantees that for every accepting leaf with path $\mathbf{r}$, the verifier output $y = s$ satisfies $s = F(\mathbf{r}) = \tilde{\mathbf{f}}(\mathbf{r})$. Since we have shown $C = \langle \mathbf{f}, \mathbf{G} \rangle$, the predicate $\varphi(\mathbf{G}, C, \mathbf{f}, \mathbf{r}) = \tilde{\mathbf{f}}(\mathbf{r})$ by definition.

Thus, for any accepting transcript, $y = \varphi(\mathbf{G}, C, \mathbf{f}, \mathbf{r})$ holds. The only way $y \notin \{\varphi(\mathbf{G}, C, \mathbf{f}, \mathbf{r}), \perp\}$ could occur is if:

- The DLOG argument fails (i.e., a non-trivial relation among random generators is found), which happens with probability at most $\mathrm{negl}(\lambda)$.
- The forking lemma fails to construct the tree despite non-negligible acceptance probability, which happens with probability at most $\mathrm{negl}(\lambda)$.

By a union bound, the total error probability is $\mathrm{negl}(\lambda)$. This establishes the $(\varphi, \mathrm{negl}(\lambda))$-knowledge attractive property for $\mathsf{BulletEval}$. □

5.4 Modular Analysis of BulletProofs

With the knowledge attractive property of $\mathsf{BulletEval}$ established, the analysis of the full BulletProofs polynomial commitment scheme follows the same modular pattern as BaseFold.

Theorem 7 (BulletProofs Security). *Under the q-discrete logarithm assumption for $q \geq 2^{\mu}$, the BulletProofs construction satisfies the security properties of Definition 3 with perfect correctness, perfect completeness, binding error $negl(\lambda)$, and knowledge soundness with knowledge error at most*

$$\varepsilon_{\mathsf{BP}}(\lambda) = \frac{2\mu}{|\mathbb{F}|} + negl(\lambda).$$

Protocol 5 (BulletProofs PCS). Let $\mathbb{G}$ be the same group as in previous discussions.

- $\mathsf{Setup}(1^\lambda, 2^\mu) \to \mathsf{pp}$: Sample $\mathbf{G} \xleftarrow{\$} \mathbb{G}^{2^\mu}$ uniformly at random. Output public parameters $\mathsf{pp} = \mathbf{G}$. The verifier parameters vp equal pp.
- $\mathsf{Commit}(\mathsf{pp}, f) \to \mathsf{cm}$: Compute the vector $\mathbf{f} = f(\{0,1\}^\mu)$. Output the Pedersen commitment $C = \langle \mathbf{f}, \mathbf{G} \rangle$.
- $\mathsf{Open}(\mathsf{pp}, C, f) \to b$: Compute $\mathbf{f}$ from f and output 1 if $C = \langle \mathbf{f}, \mathbf{G} \rangle$, and 0 otherwise.
- $\mathsf{Eval}(\mathsf{vp}, C, \mathbf{z}, y; f, \mathsf{pp}) \to b$:
  1. **Parallel execution:** Execute sum-check and $\mathsf{BulletEval}$ in parallel, sharing the same challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$:
     - **Sum-check:** Prover's input: f, $\mathbf{z}$, y; verifier's input: y, $\mathbf{z}$. The sum-check is applied to the polynomial $f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$ with claimed sum y. In each round $i \in \{0, \ldots, \mu - 1\}$, the prover sends $g_i(X_i)$, the verifier checks and samples r_i.
     - $\mathsf{BulletEval}$**:** Prover input: $\mathbf{f}$, $\mathbf{G}$. Verifier input: C, $\mathbf{G}$. In each round $i \in \{0, \ldots, \mu - 1\}$, prover sends $H_i(X)$, verifier checks and samples r_i (sharing the same r_i as in sum-check).
  2. **Final check:** Let y_{sc} be sum-check's final value and y_{be} be $\mathsf{BulletEval}$'s output. Verifier accepts if:
     - $\mathsf{BulletEval}$ does not reject (i.e., $y_{\mathsf{be}} \neq \perp$)
     - $y_{\mathsf{sc}} = y_{\mathsf{be}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$

Proof. **Correctness.** If C is generated by $\mathsf{Commit}(\mathsf{pp}, f)$, then $C = \langle \mathbf{f}, \mathbf{G} \rangle$ where $\mathbf{f} = f(\{0,1\}^\mu)$ is the evaluation vector. Hence $\mathsf{Open}(\mathsf{pp}, C, f)$ recomputes the inner product and accepts.

Completeness. Fix any multilinear polynomial f and any point $\mathbf{z} \in \mathbb{F}^\mu$, and let $C = \langle \mathbf{f}, \mathbf{G} \rangle$. In the sum-check component, the prover and verifier run sum-check on the polynomial

$$h(\mathbf{X}) := f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$$

with claimed sum $f(\mathbf{z})$. By the identity $\sum_{\mathbf{b} \in \{0,1\}^\mu} f(\mathbf{b}) \cdot \mathsf{eq}(\mathbf{b}, \mathbf{z}) = f(\mathbf{z})$, honest sum-check accepts and outputs

$$y_{\mathsf{sc}} = h(\mathbf{r}) = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}).$$

In the $\mathsf{BulletEval}$ component, since the prover uses the honest witness $\mathbf{f}$ with $C = \langle \mathbf{f}, \mathbf{G} \rangle$, the protocol accepts (the folding reduces C to $C^{(\mu)} = f(\mathbf{r}) \cdot \tilde{\mathbf{G}}(\mathbf{r})$, which passes the final check) and outputs

$$y_{\mathsf{be}} = f(\mathbf{r}).$$

Therefore the check $y_{\mathsf{sc}} = y_{\mathsf{be}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$ holds, so Eval accepts with probability 1.

Binding. Suppose $\mathsf{Open}(\mathsf{pp}, C, f_0) = \mathsf{Open}(\mathsf{pp}, C, f_1) = 1$. Then $\langle \mathbf{f}_0, \mathbf{G} \rangle = C = \langle \mathbf{f}_1, \mathbf{G} \rangle$, hence $\langle \mathbf{f}_0 - \mathbf{f}_1, \mathbf{G} \rangle = O$. If $f_0 \neq f_1$, then $\mathbf{f}_0 - \mathbf{f}_1 \neq \mathbf{0}$ forms a non-trivial discrete logarithm relation among $\mathbf{G}$, which contradicts the q-DLOG assumption. Hence the scheme is computationally binding with error at most $\mathrm{negl}(\lambda)$.

Knowledge soundness. Define the extractor $\mathcal{E}$ as the extractor from Theorem 6: on input $(\mathsf{pp}, C, \mathbf{z}, y)$, with oracle access to the prover's next-message function, it extracts a vector $\mathbf{f}$ and outputs the corresponding multilinear polynomial f. We show that the probability of verifier acceptance and $((\mathsf{pp}, C, \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}}$ is at most $\varepsilon_{\mathsf{BP}}(\lambda)$.

Let y_{sc} and y_{be} be the outputs of the sum-check and BulletEval components respectively. By the knowledge attractive property of BulletEval (Theorem 6), for the extracted f:

$$\Pr_{\mathbf{r} \xleftarrow{\$} \mathbb{F}^{\mu}, \mathbf{G} \xleftarrow{\$} \mathbb{G}^{2^{\mu}}} \left[y_{\mathsf{be}} \notin \{ f(\mathbf{r}), \perp \} \right] \leq \mathrm{negl}(\lambda).$$

On the complementary event where $y_{\mathsf{be}} \in \{f(\mathbf{r}), \perp\}$, acceptance requires $y_{\mathsf{be}} \neq \perp$, hence necessarily $y_{\mathsf{be}} = f(\mathbf{r})$ and $C = \langle \mathbf{f}, \mathbf{G} \rangle$. The assumption $((\mathsf{pp}, C, \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}}$ implies $f(\mathbf{z}) \neq y$. The final equality test then forces

$$y_{\mathsf{sc}} = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}).$$

Now sum-check is being applied to the polynomial $h(\mathbf{X}) = f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$, which has degree at most 2 in each variable, and whose claimed sum is $y \neq f(\mathbf{z})$. By Theorem 3,

$$\Pr\left[y_{\mathsf{sc}} = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}) \right] = \Pr\left[y_{\mathsf{sc}} = h(\mathbf{r}) \right] \leq \frac{2\mu}{|\mathbb{F}|}.$$

Combining the two bounds with the union bound gives

$$\Pr\left[b = 1 \wedge ((\mathsf{pp}, C, \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}} \right] \leq \frac{2\mu}{|\mathbb{F}|} + \mathrm{negl}(\lambda).$$

The knowledge error is thus $\varepsilon_{\mathsf{BP}}(\lambda) = \frac{2\mu}{|\mathbb{F}|} + \mathrm{negl}(\lambda)$. □

This modular analysis separates the cryptographic binding (handled by BulletEval's knowledge attractive property) from the algebraic soundness (handled by sum-check's repulsive property). The separation clarifies the role of each component and enables the synthesis of new protocols by substituting alternative implementations of either property.

6 Bird: A New Polynomial Commitment Scheme

The modular methodology developed in the previous sections not only simplifies the analysis of existing protocols but also enables the systematic design of new polynomial commitment schemes. This section presents Bird, a new PCS that achieves transparency, homomorphic commitments, and doubly efficient complexity (linear prover and logarithmic verifier) simultaneously. The construction arises from combining the BulletProofs commitment structure with a group-valued FRI evaluation protocol, using the functional composition methodology to ensure modular security.

6.1 Motivation: The Verifier Bottleneck in BulletProofs

The main efficiency limitation of BulletProofs-style polynomial commitments lies in the verifier's computational cost. In $\mathsf{BulletEval}$, after μ rounds of folding, the verifier must compute $\tilde{\mathbf{G}}(\mathbf{r})$ to check the final equation $C^{(\mu)} = s \cdot \tilde{\mathbf{G}}(\mathbf{r})$. This computation requires evaluating the group-valued polynomial $\tilde{\mathbf{G}}(\mathbf{X})$ at the random point $\mathbf{r}$, which costs $O(2^\mu)$ group operations.

This linear verifier cost is the bottleneck that prevents BulletProofs from achieving logarithmic verification time. The core insight behind Bird is that this bottleneck can be removed by delegating the computation of $\tilde{\mathbf{G}}(\mathbf{r})$ to the prover through a dedicated subprotocol, analogous to how BaseFold delegates the evaluation of $f(\mathbf{r})$ through FRI.

6.2 The Group-Valued FRI Protocol

To delegate the evaluation of $\tilde{\mathbf{G}}(\mathbf{r})$, we introduce a group-valued variant of the FRI protocol, denoted $\mathsf{FriEval}_{\mathbb{G}}$. The protocol extends the foldable code family $C^{(0)}, \ldots, C^{(\mu)}$ and the FRI machinery from Section 2.7 to operate on group elements, leveraging the fact that $\mathbb{G}$ is an $\mathbb{F}$-module where scalar multiplication $a \cdot G$ is defined for $a \in \mathbb{F}$ and $G \in \mathbb{G}$.

Remark 1. The Reed-Solomon encoding generalizes to group elements as follows. Let $\tilde{\mathbf{G}}(\mathbf{X})$ be a μ-variate multilinear polynomial over $\mathbb{G}$. Its coefficient vector in the standard monomial basis (distinct from the evaluation vector $\mathbf{G}$ over $\{0,1\}^\mu$) is denoted by $\mathsf{Coeff}(\tilde{\mathbf{G}})$. We encode this vector as $\mathbf{C}^{(0)} = \mathsf{Encode}_{C^{(0)}}(\mathsf{Coeff}(\tilde{\mathbf{G}}))$. The folding operation uses the same formula as standard FRI, with scalar multiplication replacing field multiplication. After μ rounds of folding with challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$, the resulting folded value equals $\tilde{\mathbf{G}}(\mathbf{r})$.

The group-valued analogue of the folding property from Section 2.7 holds: after μ rounds of folding the coefficient vector with challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$, the resulting folded value $\mathbf{C}^{(\mu)}$ equals (the repetition of) $\tilde{\mathbf{G}}(\mathbf{r})$. The Reed-Solomon encoding provides the binding property for the encoded coefficient vector.

6.3 The Bird Protocol

Bird combines the BulletProofs commitment structure with $\mathsf{FriEval}_{\mathbb{G}}$ to achieve logarithmic verification. The protocol runs three parallel components under shared randomness: a scalar sum-check, $\mathsf{BulletEval}$, and $\mathsf{FriEval}_{\mathbb{G}}$.

The key optimization compared to plain BulletProofs described in Protocol 5 is in the third acceptance condition: instead of computing $\tilde{\mathbf{G}}(\mathbf{r})$ directly (which costs $O(n)$ group operations), the verifier receives G_{fri} from the $\mathsf{FriEval}_{\mathbb{G}}$ protocol and only performs a constant-time check. This reduces the verifier's group operations from $O(n)$ to $O(\mu \cdot q)$, where $\mu = \log n$ and q is the number of FRI queries (each query requires $O(\mu)$ group operations to verify the folding chain).

The security of Bird follows from the modular composition of the three functional properties.

Protocol 6 (Group-Valued FRI ($\mathsf{FriEval}_{\mathbb{G}}$)). Let $C^{(0)}, \ldots, C^{(\mu)}$ be a foldable family of Reed-Solomon codes with $C^{(i)} = \mathsf{RS}[\mathbb{F}, L^{(i)}, \rho]$ and $|L^{(i)}| = |L^{(i-1)}|/2$. Let $N^{(i)} = |L^{(i)}|$ and $n^{(i)} = 2^{\mu-i} = \rho \cdot N^{(i)}$.

Prover input: The vector $\mathbf{G} \in \mathbb{G}^{n^{(0)}}$ viewed as the evaluation of some group-valued multilinear polynomial $\tilde{\mathbf{G}}(\mathbf{X})$ over $\{0,1\}^{\mu}$.

Verifier input: Oracle access to $\mathbf{C}^{(0)} = \mathsf{Encode}_{C^{(0)}}(\mathsf{Coeff}(\tilde{\mathbf{G}})) \in \mathbb{G}^{N^{(0)}}$.

The protocol proceeds in two phases:

Commit phase: For $i = 0, \ldots, \mu - 1$:

1. The verifier samples $r_i \xleftarrow{\$} \mathbb{F}$ and sends it to the prover.
2. The prover computes $\mathbf{C}^{(i+1)} = \mathsf{Fold}(\mathbf{C}^{(i)}, r_i)$ and sends the oracle of $\mathbf{C}^{(i+1)}$ to the verifier. If $i = \mu - 1$, the prover sends the single group element $\tilde{\mathbf{G}}(\mathbf{r})$.

Query phase: Same as standard FRI, the verifier samples a sparse matrix $\mathbf{Q}$ of size $q \times M$ where $M = \sum_{i=0}^{\mu} N^{(i)}$, and accepts if $\mathbf{Q} \cdot (\mathbf{C}^{(0)}, \ldots, \mathbf{C}^{(\mu)}) = \mathbf{0}$ (with each row containing at most 3 non-zero entries, checking the folding consistency at a random pair of positions in some round).

Theorem 8 (Bird Security). *Fix $\rho \in (0,1)$ and let $\varepsilon_{\mathsf{FRI}}$ be the attractive error of $\mathsf{FriEval}_{\mathbb{G}}$ (see below). Under the q-discrete logarithm assumption in $\mathbb{G}$ for $q \geq 2^{\mu}$, the* Bird *construction satisfies the security properties of Definition 3 with perfect correctness, perfect completeness, binding error $negl(\lambda)$, and knowledge soundness with knowledge error at most*

$$\varepsilon_{\mathsf{MW}}(\lambda) = \frac{2\mu}{|\mathbb{F}|} + \varepsilon_{\mathsf{FRI}} + negl(\lambda).$$

Proof. **Correctness.** If C is generated by $\mathsf{Commit}(\mathsf{pp}, f)$, then $C = \langle \mathbf{f}, \mathbf{G} \rangle$ by definition. Hence $\mathsf{Open}(\mathsf{pp}, C, f)$ recomputes the inner product and accepts.

Completeness. Fix any multilinear polynomial f and any point $\mathbf{z} \in \mathbb{F}^{\mu}$, and let $C = \langle \mathbf{f}, \mathbf{G} \rangle$. In the sum-check component, the prover and verifier run sum-check on the polynomial

$$h(\mathbf{X}) := f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$$

with claimed sum $f(\mathbf{z})$. By the identity $\sum_{\mathbf{b} \in \{0,1\}^{\mu}} f(\mathbf{b}) \cdot \mathsf{eq}(\mathbf{b}, \mathbf{z}) = f(\mathbf{z})$, honest sum-check accepts and outputs

$$y_{\mathsf{sc}} = h(\mathbf{r}) = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}).$$

In the $\mathsf{BulletEval}$ component, the prover uses the honest witness $\mathbf{f}$ with $C = \langle \mathbf{f}, \mathbf{G} \rangle$. The protocol folds correctly and outputs

$$y_{\mathsf{be}} = f(\mathbf{r}).$$

In the $\mathsf{FriEval}_{\mathbb{G}}$ component, $\mathbf{c}^{(0)}$ is an honestly encoded codeword of the coefficient vector of $\tilde{\mathbf{G}}$, so honest FRI accepts and outputs

$$G_{\mathsf{fri}} = \tilde{\mathbf{G}}(\mathbf{r}) \neq \perp.$$

Protocol 7 (Bird PCS). Let $C^{(0)}, \ldots, C^{(\mu)}$ be a foldable code family as defined in Section 2.7, with $C^{(i)} = \mathsf{RS}[\mathbb{F}, L^{(i)}, \rho]$, $n^{(i)} = 2^{\mu-i}$, and rate $\rho \in (0, 1)$.

- $\mathsf{Setup}(1^\lambda, 2^\mu) \to \mathsf{pp}$: Sample $\mathbf{G} \xleftarrow{\$} \mathbb{G}^{n^{(0)}}$ uniformly at random. Compute the coefficient vector of $\tilde{\mathbf{G}}(\mathbf{X})$ via MLE interpolation from $\mathbf{G}$. Compute the encoding $\mathbf{C}^{(0)} = \mathsf{Encode}_{C^{(0)}}(\mathsf{Coeff}(\tilde{\mathbf{G}}))$. Output public parameters $\mathsf{pp} = \mathbf{G}$ (the prover can optionally store the coefficient vector $\mathsf{Coeff}(\tilde{\mathbf{G}})$ and its encoding $\mathbf{C}^{(0)}$) and verifier parameters $\mathsf{vp} = [\mathbf{C}^{(0)}]$, the oracle access to $\mathbf{C}^{(0)}$.
- $\mathsf{Commit}(\mathsf{pp}, f) \to \mathsf{cm}$: Compute $\mathbf{f} = f(\{0,1\}^\mu)$. Output $C = \langle \mathbf{f}, \mathbf{G} \rangle$.
- $\mathsf{Open}(\mathsf{pp}, C, f) \to b$: Output 1 if $C = \langle \mathbf{f}, \mathbf{G} \rangle$, and 0 otherwise.
- $\mathsf{Eval}(\mathsf{vp}, C, \mathbf{z}, y; f, \mathsf{pp}) \to b$:
  1. **Parallel execution:** Execute sum-check, BulletEval, and $\mathsf{FriEval}_{\mathbb{G}}$ in parallel, sharing the same challenges $\mathbf{r} = (r_0, \ldots, r_{\mu-1})$:
     - **Sum-check:** Prover input: f, $\mathbf{z}$, y. Verifier input: y, $\mathbf{z}$. For polynomial $h(\mathbf{X}) = f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$ with claimed sum y. In each round $i \in \{0, \ldots, \mu-1\}$, verifier samples r_i. Finally, the verifier outputs y_{sc}.
     - **BulletEval:** Prover input: $\mathbf{f} = f(\{0,1\}^\mu)$, $\mathbf{G}$. Verifier input: C. In each round $i \in \{0, \ldots, \mu-1\}$, verifier samples r_i (sharing the same r_i as in sum-check). The verifier skips the final check $C^{(\mu)} = y_{\mathsf{be}} \cdot \tilde{\mathbf{G}}(\mathbf{r})$ in the original BulletEval protocol, and outputs y_{be}.
     - **$\mathsf{FriEval}_{\mathbb{G}}$:** Prover input: the vector $\mathbf{G}$, $\mathbf{C}^{(0)}$. Verifier input: oracle access to $\mathbf{C}^{(0)}$. In each round $i \in \{0, \ldots, \mu-1\}$, verifier samples r_i (sharing the same r_i as above). Finally, the verifier outputs G_{fri}.
  2. **Final check:** Verifier accepts if:
     - $\mathsf{FriEval}_{\mathbb{G}}$ does not reject (i.e., $G_{\mathsf{fri}} \neq \perp$)
     - $y_{\mathsf{sc}} = y_{\mathsf{be}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$
     - $C^{(\mu)} = y_{\mathsf{be}} \cdot G_{\mathsf{fri}}$

Therefore the final checks hold:

$$y_{\mathsf{sc}} = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}) = y_{\mathsf{be}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}),$$

$$C^{(\mu)} = f(\mathbf{r}) \cdot \tilde{\mathbf{G}}(\mathbf{r}) = y_{\mathsf{be}} \cdot G_{\mathsf{fri}},$$

so Eval accepts with probability 1.

Binding. Suppose $\mathsf{Open}(\mathsf{pp}, C, f_0) = \mathsf{Open}(\mathsf{pp}, C, f_1) = 1$. Then $\langle \mathbf{f}_0, \mathbf{G} \rangle = C = \langle \mathbf{f}_1, \mathbf{G} \rangle$, hence $\langle \mathbf{f}_0 - \mathbf{f}_1, \mathbf{G} \rangle = O$. If $f_0 \neq f_1$, then $\mathbf{f}_0 - \mathbf{f}_1 \neq \mathbf{0}$ forms a non-trivial discrete logarithm relation for $\mathbf{G}$, which contradicts the q-DLOG assumption for $q = 2^\mu$. Hence Bird is computationally binding with error at most $\mathrm{negl}(\lambda)$.

Knowledge soundness. Define the extractor $\mathcal{E}$ as the extractor from Theorem 6: on input $(\mathsf{pp}, C, \mathbf{z}, y)$, with oracle access to the prover's next-message function, it extracts a coefficient vector $\mathbf{f}$ and outputs the corresponding multilinear polynomial f. We show that the probability of verifier acceptance and $((\mathsf{pp}, C, \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}}$ is at most $\varepsilon_{\mathsf{MW}}(\lambda)$.

Let y_{sc}, y_{be}, and G_{fri} be the outputs of the three protocol components respectively. By the knowledge attractive property of BulletEval (Theorem 6), for the

extracted f:

$$\Pr_{\mathbf{r}\xleftarrow{\$}\mathbb{F}^{\mu},\mathbf{G}\xleftarrow{\$}\mathbb{G}^{2^{\mu}}}\big[y_{\mathsf{be}} \notin \{f(\mathbf{r}), \perp\}\big] \leq \mathrm{negl}(\lambda),$$

and moreover $C = \langle \mathbf{f}, \mathbf{G} \rangle$ on this high-probability event.

By the attractive property of $\mathsf{FriEval}_{\mathbb{G}}$, which follows from the same folding consistency analysis as field-valued FRI (Theorem 4) applied to the group-valued code, there exists an error bound $\varepsilon_{\mathsf{FRI}}$ such that

$$\Pr\big[G_{\mathsf{fri}} \notin \{\tilde{\mathbf{G}}(\mathbf{r}), \perp\}\big] \leq \varepsilon_{\mathsf{FRI}}.$$

On the intersection of these two high-probability events, acceptance requires $y_{\mathsf{be}} \neq \perp$ and $G_{\mathsf{fri}} \neq \perp$, hence necessarily $y_{\mathsf{be}} = f(\mathbf{r})$ and $G_{\mathsf{fri}} = \tilde{\mathbf{G}}(\mathbf{r})$. The assumption $((\mathsf{pp}, C, \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}}$ implies $f(\mathbf{z}) \neq y$. The first final check $y_{\mathsf{sc}} = y_{\mathsf{be}} \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})$ then forces

$$y_{\mathsf{sc}} = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z}).$$

The second final check $C^{(\mu)} = y_{\mathsf{be}} \cdot G_{\mathsf{fri}}$ is automatically satisfied since $C^{(\mu)} = f(\mathbf{r}) \cdot \tilde{\mathbf{G}}(\mathbf{r})$ follows from the $\mathsf{BulletEval}$ knowledge attractive guarantee.

Now sum-check is being applied to the polynomial $h(\mathbf{X}) = f(\mathbf{X}) \cdot \mathsf{eq}(\mathbf{X}, \mathbf{z})$, which has degree at most 2 in each variable, and whose claimed sum is $y \neq f(\mathbf{z})$. By Theorem 3,

$$\Pr\big[y_{\mathsf{sc}} = f(\mathbf{r}) \cdot \mathsf{eq}(\mathbf{r}, \mathbf{z})\big] = \Pr\big[y_{\mathsf{sc}} = h(\mathbf{r})\big] \leq \frac{2\mu}{|\mathbb{F}|}.$$

Combining the three bounds with the union bound gives

$$\Pr\big[b = 1 \wedge ((\mathsf{pp}, C, \mathbf{z}, y), f) \notin \mathcal{R}_{\mathsf{Eval}}\big] \leq \frac{2\mu}{|\mathbb{F}|} + \varepsilon_{\mathsf{FRI}} + \mathrm{negl}(\lambda).$$

The knowledge error is thus $\varepsilon_{\mathsf{MW}}(\lambda) = \frac{2\mu}{|\mathbb{F}|} + \varepsilon_{\mathsf{FRI}} + \mathrm{negl}(\lambda)$. □

6.4 Efficiency

Theorem 9 (Bird Efficiency). *After instantiating the oracle with Merkle roots, the* Bird *protocol for a polynomial of degree $n = 2^{\mu}$ has:*

- ***Setup cost:*** *$O(n \log n)$ group operations.*
- ***Commit cost:*** *$O(n)$ group operations, or $O(n/\log n)$ field operations using multi-scalar multiplication algorithm.*
- ***Prover cost:*** *$O(n)$ field operations and $O(n)$ group operations.*
- ***Verifier cost:*** *$O(\mu^2 \cdot q)$ hash operations, $O(\mu \cdot q)$ group operations, and $O(\mu)$ field operations.*
- ***Proof size:*** *$O(\mu^2 \cdot q)$ hash digests and $O(\mu)$ group and field elements.*
- ***Round complexity:*** *μ rounds.*

Here $\mu = \log n$ and $q = O(\lambda)$ is the number of FRI queries.

Proof. **Setup.** The setup algorithm samples $\mathbf{G} \xleftarrow{\$} \mathbb{G}^n$ uniformly at random, costing $O(n)$ group operations. It then computes the coefficient vector of $\tilde{\mathbf{G}}(\mathbf{X})$ via MLE interpolation from the evaluation vector $\mathbf{G}$. This inverse MLE transform costs $O(n \log n)$ group operations (similar to an inverse Walsh-Hadamard transform). Finally, it computes the Reed-Solomon encoding $\mathbf{C}^{(0)} = \mathsf{Encode}_{C^{(0)}}(\mathsf{Coeff}(\tilde{\mathbf{G}}))$, which requires an FFT of size n/ρ, costing $O(n \log n)$ group operations. The total setup cost is therefore $O(n \log n)$ group operations.

Commit. Suppose f is provided in its evaluation form, which is common in practice, then the commitment generation is simply computing the Pedersen commitment $C = \langle \mathbf{f}, \mathbf{G} \rangle$, a multi-scalar multiplication of size n costing $O(n)$ group operations or $O(n/\log n)$ field operations.

Prover. The prover's work is dominated by the folding operations in each round. In round i, the prover folds vectors of size $n/2^{i-1}$, costing $O(n/2^{i-1})$ operations (field operations for sum-check, group operations for $\mathsf{BulletEval}$ and $\mathsf{FriEval}_{\mathbb{G}}$). Summing over all rounds gives $O(n)$ total field/group operations.

Verifier cost. The verifier's work consists of: (a) generating random challenges (μ field operations), (b) checking the final equations ($O(1)$ operations), and (c) the query phase. The query phase requires q queries, each requiring verification of $O(\mu)$ Merkle tree levels. Since there are μ Merkle trees (one per round), the total hash operations per query is $\sum_{i=1}^{\mu} O(\mu - i) = O(\mu^2)$, giving $O(\mu^2 \cdot q)$ total. Group operations for folding verification cost $O(\mu \cdot q)$.

Proof size. The proof consists of: the prover's cross-term commitments from $\mathsf{BulletEval}$ ($O(\mu)$ group elements), the Merkle proofs for q queries at each of μ rounds ($O(\mu^2 \cdot q)$ hash digests), and the final scalars. The total size is $O(\mu^2 \cdot q)$ hash digests plus $O(\mu)$ group and field elements.

Round complexity. All three protocols run in parallel with shared randomness, so the total round complexity is μ rounds. □

Note that compared to BaseFold, the cost of encoding $\mathbf{G}$, which is quasilinear, is paid only once at setup time, thus achieving both linear commitment generation cost and linear proving cost. In summary, Bird achieves three desirable properties simultaneously:

- **Transparency:** The setup requires only random generators, no trusted setup. This follows from the BulletProofs commitment structure.
- **Homomorphism:** Commitments are additive due to the linear structure of Pedersen commitments. For two polynomials f_1, f_2 with commitments C_1, C_2, the commitment to $f_1 + f_2$ is $C_1 + C_2$.
- **Doubly efficient:** The prover has $O(n)$ complexity and the verifier has logarithmic level $O(\lambda \cdot \log^2 n)$ complexity.

Table 1 compares Bird with existing polynomial commitment schemes.

Compared to the concurrent work of Eagen et al. [15], which also achieves transparency, homomorphism, and double efficiency through a sequential composition of BulletProofs and group-valued BaseFold, Bird uses a parallel interleaved design that reduces round complexity by half and directly composes with FRI rather than introducing an additional sum-check layer.

Scheme	Commit	Prover	Verifier	Transp.	Homom.
KZG [21]	$O(n \log n)$	$O(n \log n)$	$O(1)$	No	Yes
BulletProofs [11]	$O(n)$	$O(n)$	$O(n)$	Yes	Yes
BaseFold [45]	$O(n \log n)$	$O(n)$	$O(\mu \cdot q)$	Yes	No
Eagen et al. [15]	$O(n)$	$O(n)$	$O(\mu \cdot q)$	Yes	Yes
Bird (this work)	$O(n)$	$O(n)$	$O(\mu \cdot q)$	Yes	Yes

Table 1. Comparison of polynomial commitment schemes for multilinear polynomials with $n = 2^\mu$ coefficients. Here $q = O(\lambda / \log(1/\rho))$ is the number of FRI queries for security parameter λ and code rate ρ. "Transp." indicates transparency (no trusted setup), and "Homom." indicates homomorphic commitments.

7 Conclusion

We introduced a functional perspective that characterizes sum-check-based protocols through repulsive and attractive verifier properties. This modular, composable viewpoint captures the intrinsic security of sum-check and FRI, yielding simpler, more transparent analyses than monolithic arguments. We demonstrated the methodology's power via three applications: modular analyses of BaseFold and BulletProofs, and the construction of Bird, a transparent, homomorphic polynomial commitment scheme. Decomposing protocols into independent functional components promises broader applicability to proof systems that interleave multiple subprotocols.

Future work should extend the attractive property to larger proximity bounds, approaching the list-decoding radius to strengthen theoretical foundations; apply the decomposition methodology to other sum-check-based constructions such as Spartan for further simplification; and explore analogues of the repulsive/attractive duality in other cryptographic primitives to uncover new structural insights.

References

1. Arnon, G., Chiesa, A., Fenzi, G., Yogev, E.: Stir: Reed–solomon proximity testing with fewer queries (2024), https://eprint.iacr.org/2024/390
2. Arnon, G., Chiesa, A., Fenzi, G., Yogev, E.: Whir: Reed–solomon proximity testing with super-fast verification (2024), https://eprint.iacr.org/2024/1586
3. Aztec, T.: Aztec (2022), https://zk.money/
4. Bagad, S., Dao, Q., Domb, Y., Thaler, J.: Speeding up sum-check proving (2025), https://eprint.iacr.org/2025/1117
5. Baweja, A., Chiesa, A., Fedele, E., Fenzi, G., Mishra, P., Mopuri, T., Zitek-Estrada, A.: Time-space trade-offs for sumcheck (2025), https://eprint.iacr.org/2025/1473
6. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M., Chatzigiannakis, I., Kaklamanis, C., Marx, D., Sannella, D.: Fast reed-solomon interactive oracle proofs of proximity. In: 45th International Colloquium on Automata, Languages, and Programming (ICALP 2018). vol. 107, p. 14:1–14:17. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2018). https://doi.org/10.4230/LIPIcs.ICALP.2018.14, http://drops.dagstuhl.de/opus/volltexte/2018/9018

7. Ben-Sasson, E., Goldberg, L., Kopparty, S., Saraf, S.: Deep-fri: Sampling outside the box improves soundness (2019), https://eprint.iacr.org/2019/336
8. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference on - ITCS '12. pp. 326–349. ACM Press, Cambridge, Massachusetts (2012). https://doi.org/10.1145/2090236.2090263, http://dl.acm.org/citation.cfm?doid=2090236.2090263
9. Bootle, J., Chiesa, A., Hu, Y., Orrù, M.: Gemini: Elastic snarks for diverse environments (2022), http://eprint.iacr.org/2022/420
10. Bootle, J., Chiesa, A., Sotiraki, K.: Sumcheck arguments and their applications (2021), http://eprint.iacr.org/2021/333
11. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA. p. 315–334. IEEE Computer Society (2018). https://doi.org/10.1109/SP.2018.00020
12. Bünz, B., Fisch, B., Szepieniec, A., Canteaut, A., Ishai, Y.: Transparent snarks from dark compilers. In: Advances in Cryptology – EUROCRYPT 2020. pp. 677–706. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-45721-1_24
13. Chen, B., Bünz, B., Boneh, D., Zhang, Z.: Hyperplonk: Plonk with linear-time prover and high-degree custom gates (2022), https://eprint.iacr.org/2022/1355
14. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N., Canteaut, A., Ishai, Y.: Marlin: Preprocessing zksnarks with universal and updatable srs. In: Advances in Cryptology – EUROCRYPT 2020. pp. 738–768. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-45721-1_26
15. Eagen, L., Gabizon, A.: Revisiting the ipa-sumcheck connection (2025), https://eprint.iacr.org/2025/1325
16. Fischlin, M., Coron, J.S., Groth, J.: On the size of pairing-based non-interactive arguments (2016). https://doi.org/10.1007/978-3-662-49896-5-11, http://link.springer.com/10.1007/978-3-662-49896-5-11
17. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge (2019), https://eprint.iacr.org/2019/953
18. Ganesh, C., Patranabis, S., Singh, N.: Samaritan: linear-time prover snark from new multilinear polynomial commitments (2025), https://eprint.iacr.org/2025/419
19. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: interactive proofs for muggles. In: Proceedings of the fourtieth annual ACM symposium on Theory of computing - STOC 08. p. 113. ACM Press, Victoria, British Columbia, Canada (2008). https://doi.org/10.1145/1374376.1374396, http://dl.acm.org/citation.cfm?doid=1374376.1374396
20. Golovnev, A., Lee, J., Setty, S., Thaler, J., Wahby, R.S.: Brakedown: Linear-time and post-quantum snarks for r1cs (2021), https://eprint.iacr.org/2021/1043
21. Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Rangan, C.P., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Abe, M., Kate, A., Zaverucha,

G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications (2010). https://doi.org/10.1007/978-3-642-17373-8_11, http://link.springer.com/10.1007/978-3-642-17373-8_11
22. Ishai, Y., Rijmen, V., Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for r1cs (2019). https://doi.org/10.1007/978-3-030-17653-2_4, http://link.springer.com/10.1007/978-3-030-17653-2_4
23. Kohrita, T., Towa, P.: Zeromorph: Zero-knowledge multilinear-evaluation proofs from homomorphic univariate commitments (2023), https://eprint.iacr.org/2023/917
24. Kothapalli, A., Setty, S.: Hypernova: Recursive arguments for customizable constraint systems (2023), https://eprint.iacr.org/2023/573
25. Lee, J.: Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments (2020), http://eprint.iacr.org/2020/1274
26. Liu, T., Xie, X., Zhang, Y., Kim, Y., Kim, J., Vigna, G., Shi, E.: zkcnn: zero knowledge proofs for convolutional neural network predictions and accuracy. In: CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021. p. 2968–2985. ACM (2021). https://doi.org/10.1145/3460120.3485379
27. Liu, T., Zhang, Z., Zhang, Y., Hu, W., Zhang, Y.: Ceno: non-uniform, segment and parallel zero-knowledge virtual machine. Journal of Cryptology **38**(2), 17 (2025). https://doi.org/10.1007/s00145-024-09533-2, https://doi.org/10.1007/s00145-024-09533-2
28. Loopring, T.: Loopring - zkrollup layer2 for trading and payment (2022), https://loopring.org/
29. Lund, C., Fortnow, L., Karloff, H.J., Nisan, N.: Algebraic methods for interactive proof systems. In: 31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I. p. 2–10. IEEE Computer Society (1990). https://doi.org/10.1109/FSCS.1990.89518
30. Maller, M., Bowe, S., Kohlweiss, M., Meiklejohn, S., Cavallaro, L., Kinder, J., Wang, X., Katz, J.: Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019. p. 2111–2128. ACM (2019). https://doi.org/10.1145/3319535.3339817
31. Oswald, E., Fischlin, M., Chiesa, A., Tromer, E., Virza, M.: Cluster computing in zero knowledge (2015). https://doi.org/10.1007/978-3-662-46803-6_13, http://link.springer.com/10.1007/978-3-662-46803-6_13
32. Papini, S., Haböck, U.: Improving logarithmic derivative lookups using gkr (2023), https://eprint.iacr.org/2023/1284
33. Parno, B., Howell, J., Gentry, C., Raykova, M.: Pinocchio: Nearly practical verifiable computation. In: 2013 IEEE Symposium on Security and Privacy. pp. 238–252. IEEE, Berkeley, CA (2013). https://doi.org/10.1109/SP.2013.47, http://ieeexplore.ieee.org/document/6547113/
34. Qu, W., Sun, Y., Liu, X., Lu, T., Guo, Y., Chen, K., Zhang, J.: zkgpt: an efficient non-interactive zero-knowledge proof framework for llm inference (2025), https://eprint.iacr.org/2025/1184
35. Ràfols, C., Zapico, A.: An algebraic framework for universal and updatable snarks. In: Annual International Cryptology Conference. p. 774–804. Springer (2021)
36. Scroll, T.: Scroll (2022), https://scroll.io/

37. Setty, S.: Spartan: Efficient and general-purpose zksnarks without trusted setup. In: Annual International Cryptology Conference. p. 704–737. Springer (2020)
38. Setty, S., Thaler, J., Wahby, R.: Unlocking the lookup singularity with lasso (2023), https://eprint.iacr.org/2023/1216
39. Sun, H., Li, J., Zhang, H., Luo, B., Liao, X., Xu, J., Kirda, E., Lie, D.: zkllm: zero knowledge proofs for large language models. In: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024. p. 4405–4419. ACM (2024). https://doi.org/10.1145/3658644.3670334
40. W, W.: Folding-based zkllm (2024), https://eprint.iacr.org/2024/480
41. Wahby, R.S., Tzialla, I., Shelat, A., Thaler, J., Walfish, M.: Doubly-efficient zksnarks without trusted setup. In: 2018 IEEE Symposium on Security and Privacy (SP). pp. 926–943 (2018). https://doi.org/10.1109/SP.2018.00060
42. Walfish, M., Blumberg, A.J.: Verifying computations without reexecuting them. Communications of the ACM **58**(2), 74–84 (2015). https://doi.org/10.1145/2641562, https://dl.acm.org/doi/10.1145/2641562
43. Xie, T., Zhang, J., Zhang, Y., Papamanthou, C., Song, D., Boldyreva, A., Micciancio, D.: Libra: Succinct zero-knowledge proofs with optimal prover computation. In: Advances in Cryptology – CRYPTO 2019. pp. 733–764. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-26954-8-24
44. Xie, T., Zhang, Y., Song, D.: Orion: Zero knowledge proof with linear prover time. Cryptology ePrint Archive (2022), https://eprint.iacr.org/2022/1010
45. Zeilberger, H., Chen, B., Fisch, B.: Basefold: Efficient field-agnostic polynomial commitment schemes from foldable codes (2023), https://eprint.iacr.org/2023/1705
46. Zhang, J., Xie, T., Zhang, Y., Song, D.: Transparent polynomial delegation and its applications to zero knowledge proof. In: 2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020. p. 859–876. IEEE (2020). https://doi.org/10.1109/SP40000.2020.00052
47. Zhang, Y., Szepeniec, A., Zhang, R., Sun, S.F., Wang, G., Gu, D.: Voproof: Efficient zksnarks from vector oracle compilers. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. p. 3195–3208. Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3548606.3559387, https://doi.org/10.1145/3548606.3559387
48. zkSync, T.: zksync (2022), https://zksync.io/