A Simplified Round-by-round Soundness Proof of FRI

Albert Garreta* Nicolas Mohnblatt† Benedikt Wagner‡

November 4, 2025

Abstract

The FRI protocol (ICALP '18) is one of the most influential and widely deployed building blocks at the core of modern SNARK systems. While its concrete security is well understood, existing security proofs are intricate and technically complex.

In this work, we present a significantly simpler security analysis of FRI, in particular its round-by-round soundness. Our approach is more accessible to a broader audience, lowering the barrier to understanding this fundamental protocol. Furthermore, the simplicity of our analysis may pave the way for future formal verification efforts of modern SNARK constructions.

Contents

*Nethermind Research, albert@nethermind.io
†zkSecurity, nico@zksecurity.xyz
‡Ethereum Foundation, benedikt.wagner@ethereum.org

1 Introduction

The FRI protocol [BBHR18] is a foundational component of modern cryptographic argument systems, providing an efficient method for proving proximity to Reed–Solomon (RS) codes. It lies at the core of an entire industry of succinct hash-based arguments in production. To highlight this importance, note that this industry will play a critical role in securing billions of dollars on Ethereum in the future[1]. Given its central importance, it is crucial to have a precise and thorough understanding of FRI's security guarantees.

While several analyses of FRI exist [BCI+20, Sta21, BGK+23, ABN23, AFR23, EA23] and its concrete security is well understood, the existing proofs are technically complex and can be challenging to follow. In this work, we present a *simpler and more accessible proof* of round-by-round soundness for FRI, which is the notion needed to deduce security of the non-interactive protocol that is built from it [BCS16, CCH+19, Hol19]. This new analysis offers two main benefits:

- *Accessibility for new researchers*: By lowering the technical entry barrier, we aim to make this area more approachable, encouraging new contributions and fresh ideas that could shape the future of the field.
- *A foundation for formal verification:* Our simplified analysis can serve as a template for formal verification efforts, which are increasingly important in the context of production SNARK systems.

In particular, one of the ambitious goals of Ethereum's post-quantum roadmap is to achieve formal verification of every component in the stack[2]. We believe that our work contributes to this vision by making the security analysis of a key protocol more tractable.

We note that our proof relies on the *mutual correlated agreement (MCA) property* [ACY23, ACFY24b], whereas standard analyses of FRI can be carried out under the weaker correlated agreement property. Since MCA is a plausible conjecture believed by most domain experts, and our contribution lies in simplifying an existing result rather than proving a new one, we view this as a reasonable trade-off.

[1] See for instance https://ethproofs.org/.
[2] See https://verified-zkevm.org/.

2 Preliminaries

Most of the notation and preliminaries in this section are lifted almost verbatim from STIR and WHIR [ACFY24a, ACFY24b]. As a quick reference, we use the following notation:

- The notation $x \leftarrow X$ means that x is sampled uniformly at random from a finite set X.
- The "hat" symbol over a function (*e.g.*, $\widehat{p}$) denotes that it is a polynomial.
- Given a polynomial $\widehat{p}$, we write $\mathsf{deg}_X(\widehat{p})$ to denote the degree of $\widehat{p}$ in the variable X. We omit the subscript when $\widehat{p}$ is a univariate polynomial.
- For two functions $f, g \colon \mathcal{L} \to \Sigma$, $\Delta(f, g)$ denotes the fractional Hamming distance between f and g, *i.e.*, the fraction of points in which they disagree.
- For a set $\mathcal{S} \subseteq \Sigma^n$, $\Delta(f, \mathcal{S}) := \min_{h \in \mathcal{S}} \Delta(f, h)$.
- For a function $f \colon \mathcal{L} \to \Sigma$ and a set $S \subseteq \mathcal{L}$, we write $f[S]$ to denote the restriction of f to the set S. In particular, for two such functions f, g, we have

$$f[S] = g[S] \iff \forall s \in S : \ f(s) = g(s).$$

- For a ternary relation $\mathcal{R} = \{(\mathtt{x}, \mathtt{y}, \mathtt{w})\}$, let $L(\mathcal{R}) := \{(\mathtt{x}, \mathtt{y}) \mid \exists \mathtt{w}, (\mathtt{x}, \mathtt{y}, \mathtt{w}) \in \mathcal{R}\}$ be the language induced by $\mathcal{R}$.
- A set $\mathcal{L} \subseteq \mathbb{F}$ is *m-smooth* if it is a multiplicative coset of $\mathbb{F}^*$ whose order is a power of m.
- For a set $\mathcal{L} \subseteq \mathbb{F}$ and $m \in \mathbb{N}$, we define $\mathcal{L}^m := \{x^m : x \in \mathcal{L}\}$.
- We assume that all sets in this work admit some canonical ordering of their elements.

2.1 Reed-Solomon Codes

FRI allows to prove proximity to a Reed-Solomon code. We recall Reed-Solomon codes here. In general, a *linear code* of length n over a field $\mathbb{F}$ is a linear subspace $\mathcal{C} \subseteq \mathbb{F}^n$. In the context of FRI and Reed-Solomon codes, it is more convenient to interpret a codeword $f \in \mathcal{C}$ as the evaluations of a univariate polynomial $\widehat{f} \in \mathbb{F}^{<d}[X]$ over a fixed domain of size n. We write codewords $f \in \mathcal{C}$ as functions $f : \mathcal{L} \to \mathbb{F}$, where we assume that $\mathcal{L}$ has some implicit canonical ordering.

Definition 2.1 (Reed-Solomon code)**.** The *Reed-Solomon code* over field $\mathbb{F}$, evaluation domain $\mathcal{L} \subseteq \mathbb{F}$, and degree $d \in \mathbb{N}$ is the set of evaluations over $\mathcal{L}$ of univariate polynomials (over $\mathbb{F}$) of degree less than d:

$$\mathsf{RS}[\mathbb{F}, \mathcal{L}, d] := \left\{ f \colon \mathcal{L} \to \mathbb{F} \quad : \quad \exists \widehat{f} \in \mathbb{F}^{<d}[X], \ \forall x \in \mathcal{L}, \ f(x) = \widehat{f}(x) \right\}.$$

The *rate* of $\mathsf{RS}[\mathbb{F}, \mathcal{L}, d]$ is $\rho := d/|\mathcal{L}|$. Its *dimension* is d. The *minimum distance* is $1 - \rho$.

2.2 Mutual correlated agreement for proximity generators

Intuitively, mutual correlated agreement states that if the linear combination $\sum_j r_j f_j$ for some functions f_j and random coefficients r_j agrees with a codeword on a sufficiently large domain S, then each of the f_j agrees with a codeword on *the very same* domain S – all of that with high probability over the r_j. The definition below assumes that r_j are sampled by an algorithm Gen, which allows to model different distributions of r_j's. We note that mutual correlated agreement can be defined for general codes, but we restrict ourselves to Reed-Solomon codes as this is sufficient for this work.

Definition 2.2 (Proximity generator, mutual correlated agreement)**.** Let $\mathcal{C} := \mathsf{RS}[\mathbb{F}, \mathcal{L}, d]$ be a Reed-Solomon code with rate ρ. We say that Gen is a proximity generator for $\mathcal{C}$ with mutual

correlated agreement with proximity bound $\mathsf{B}^\star$ and error $\mathsf{err}^\star$ if for every $f_1, \ldots, f_m \colon \mathcal{L} \to \mathbb{F}$ and every $\delta \in [0, 1 - \mathsf{B}^\star(\rho))$ the following holds:

$$\Pr_{(r_1,\ldots,r_m)\leftarrow\mathsf{Gen}(m)}\left[\exists S \subseteq \mathcal{L}\ s.t.\ \begin{array}{c} |S| \geq (1-\delta)\cdot n \\ \wedge\ \exists u \in \mathcal{C}, u[S] = \sum_{j\in[m]} r_j \cdot f_j[S] \\ \wedge\ \exists i \in [m], \forall u' \in \mathcal{C}, u'[S] \neq f_i[S] \end{array}\right] \leq \mathsf{err}^\star(\mathcal{C}, m, \delta).$$

Remark 2.3. Notice that in the definition above, we allow $\delta = 0$, in contrast to previous works [ACY23, ACFY24b] which specify $\delta \in (0, 1 - \mathsf{B}^\star)$. This change is of no consequence to existing results on proximity generators. Indeed, $\delta = 0$ corresponds to the case $S = [n]$; a case also covered by setting $\delta \in (0, 1/n)$. We find that including $\delta = 0$ simplifies our proof by removing edge cases.

Definition 2.4 (Powers proximity generator)**.** Let $m \in \mathbb{N}$ be a length parameter and $r \in \mathbb{F}$ be a field element. We define the Powers proximity generator as the following randomized algorithm, taking a uniform $r \in \mathbb{F}$ as input:

$$\mathsf{Powers}(m; r) := (1, r, \ldots, r^{m-1}) \in \mathbb{F}^m.$$

Lemma 2.5 (cf. [ACFY24b])**.** *Let $\mathcal{C} := \mathsf{RS}[\mathbb{F}, \mathcal{L}, d]$ be a Reed-Solomon code with rate ρ. The function* Powers *defined in Definition 2.4 is a proximity generator with mutual correlated agreement with proximity bound $\mathsf{B}^\star(\rho)$ and error $\mathsf{err}^\star(\mathcal{C}, m, \delta)$, where*

$$\mathsf{B}^\star(\rho) = \frac{1+\rho}{2} \quad \text{and} \quad \mathsf{err}^\star(\mathcal{C}, m, \delta) = \frac{(m-1)\cdot d}{\rho|\mathbb{F}|}.$$

Conjecture 2.6. *The function* Powers *defined in Definition 2.4 is a proximity generator with mutual correlated agreement for every smooth Reed-Solomon code $\mathcal{C} := \mathsf{RS}[\mathbb{F}, \mathcal{L}, d]$ (with rate $\rho := d/|\mathcal{L}|$). We give two conjectures, for the parameters of the proximity bound $\mathsf{B}^\star$ and the error $\mathsf{err}^\star$:*

1. *Up to the Johnson bound: $\mathsf{B}^\star(\rho) := \sqrt{\rho}$, and*

$$\mathsf{err}^\star(\mathcal{C}, m, \delta) := \frac{(m-1)\cdot d^2}{|\mathbb{F}| \cdot \left(2 \cdot \min\left\{1 - \sqrt{\rho} - \delta, \frac{\sqrt{\rho}}{20}\right\}\right)^7}.$$

2. *Up to capacity: $\mathsf{B}^\star(\rho) := \rho$, and there exists constants $c_1, c_2 \in \mathbb{N}$ such that for every $\eta > 0$ and $0 < \delta < 1 - \rho - \eta$:*

$$\mathsf{err}^\star(\mathcal{C}, m, \delta) := \frac{(m-1)^{c_2} \cdot d^{c_2}}{\eta^{c_1} \cdot \rho^{c_1+c_2} \cdot |\mathbb{F}|}.$$

3 IOPPs and Round-By-Round Soundness

Before we move our attention to FRI and its new analysis, we recall the notion of IOPPs and round-by-round soundness.

3.1 Interactive oracle proof of proximity

In an *interactive oracle proof* for a relation $\mathcal{R}$, we consider a prover and a verifier, where the prover wants to convince the verifier of the validity of some statement. To this end, the prover can provide oracles to the verifier and the verifier can respond with challenges. The verifier can additionally query the oracles provided by the prover. For convenience, we split the statement into $\mathbb{x}$, to which the verifier has full access, and into an oracle part $\mathbb{y}$, to which the verifier has oracle access. The prover also knows some witness $\mathbb{w}$. Completeness states that the honest prover convinces the verifier if $(\mathbb{x}, \mathbb{y}, \mathbb{w}) \in \mathcal{R}$. Soundness instead bounds the probability of the prover succeeding if $(\mathbb{x}, \mathbb{y})$ is a no-instance. For an *interactive oracle proof of proximity* (IOPP), it is convenient to assume that $\mathbb{x}$ describes the parameters of a code, and $\mathbb{y}$ describes an alleged codeword, and the soundness bound should depends only on the (Hamming) distance of the provided $\mathbb{y}$ to the code specified by $\mathbb{x}$. For this work, it is sufficient to consider *public-coin* IOPPs, where the verifier's messages are all uniformly sampled from prescribed sets (that are statistically independent from the execution of the protocol). In this case, we denote $\mathbf{V}^{\mathbb{y},\pi_1,\ldots,\pi_{\mathsf{k}}}(\mathbb{x}, \alpha_1, \ldots, \alpha_{\mathsf{k}+1})$ to be the deterministic function that has oracle access to $\mathbb{y}$ and the prover messages, takes as input the verifier messages and outputs the verifier's decision bit. We now give the precise definition.

Definition 3.1 (Interactive oracle proof of proximity)**.** We say that $\mathsf{IOPP} = (\mathbf{P}, \mathbf{V})$ is a public-coin IOP of proximity for a relation $\mathcal{R}$ with k rounds, perfect completeness and soundness β if the following holds:

- **Perfect completeness.** For every $(\mathbb{x}, \mathbb{y}, \mathbb{w}) \in \mathcal{R}$,

$$\Pr_{\alpha_1,\ldots,\alpha_{\mathsf{k}}}\left[\begin{array}{c|r} \mathbf{V}^{\mathbb{y},\pi_1,\ldots,\pi_{\mathsf{k}}}(\mathbb{x}, \alpha_1, \ldots, \alpha_{\mathsf{k}}) = 1 & \pi_1 \leftarrow \mathbf{P}(\mathbb{x}, \mathbb{y}, \mathbb{w}, \alpha_1) \\ & \vdots \\ & \pi_{\mathsf{k}} \leftarrow \mathbf{P}(\mathbb{x}, \mathbb{y}, \mathbb{w}, \alpha_1, \ldots, \alpha_{\mathsf{k}}) \end{array}\right] = 1.$$

- **Soundness.** For every $(\mathbb{x}, \mathbb{y}) \notin L(\mathcal{R})$ and unbounded malicious prover $\tilde{\mathbf{P}}$,

$$\Pr_{\alpha_1,\ldots,\alpha_{\mathsf{k}}}\left[\begin{array}{c|r} \mathbf{V}^{\mathbb{y},\pi_1,\ldots,\pi_{\mathsf{k}}}(\mathbb{x}, \alpha_1, \ldots, \alpha_{\mathsf{k}}) = 1 & \pi_1 \leftarrow \tilde{\mathbf{P}}(\alpha_1) \\ & \vdots \\ & \pi_{\mathsf{k}} \leftarrow \tilde{\mathbf{P}}(\alpha_1, \ldots, \alpha_{\mathsf{k}}) \end{array}\right] \leq \beta(\mathbb{x}, \mathbb{y}).$$

  When the soundness error depends only on the lengths of the inputs and on the distance δ of $\mathbb{y}$ from the language $L_{\mathbb{x}} := \{\mathbb{y}' : \exists \mathbb{w}, (\mathbb{x}, \mathbb{y}', \mathbb{w}) \in \mathcal{R}\}$, we write $\beta(|\mathbb{x}|, |\mathbb{y}|, \delta)$ (and sometimes leave out $|\mathbb{x}|$ and $|\mathbb{y}|$, writing $\beta(\delta)$, when the lengths are clear from context).

Remark 3.2 (Verifier moves last)**.** The above definition assumes that the prover moves last and sends the final message π_{k}. In most IOPPs for codes (*e.g.* FRI [BBHR18], BaseFold [ZCF24], STIR [ACFY24a], WHIR [ACFY24b]), it is the verifier that moves last. In such cases, we set $\pi_{\mathsf{k}} = \perp$ and omit it from our notation.

Remark 3.3 (IOPP for Reed-Solomon codes)**.** Let $\mathcal{C}$ be a Reed-Solomon code. We say that $(\mathbf{P}, \mathbf{V})$ is an *IOPP for* $\mathcal{C}$ if it is an IOPP for the ternary relation $\mathcal{R}_{\mathcal{C}} = \{(\mathcal{C}, f, \perp) \mid f \in \mathcal{C}\}$. When it is clear from context, we omit $\mathcal{C}$ from our notation, writing for example $\mathsf{State}(f, \emptyset)$ instead of $\mathsf{State}(\mathcal{C}, f, \emptyset)$ or $\mathbf{V}^{f,\pi_1,\ldots,\pi_{\mathsf{k}}}(\alpha_1, \ldots, \alpha_{\mathsf{k}})$ instead of $\mathbf{V}^{\mathbb{y},\pi_1,\ldots,\pi_{\mathsf{k}}}(\mathbb{x}, \alpha_1, \ldots, \alpha_{\mathsf{k}})$.

3.2 Round-by-round soundness

We recall the notion of round-by-round soundness [CCH+19]. Before we can define it, we first define what a state function is. Intuitively, it associates every (partial) transcript with a state that is either 0 or 1. Intuitively, a 0-state indicates that the prover is about to lose, *i.e.*, the verifier is about to reject. A malicious prover should start with state 0, and if in the end of the interaction the state is still 0, then the verifier rejects.

Definition 3.4 (State function)**.** Let $(\mathbf{P}, \mathbf{V})$ be an IOPP for a relation $\mathcal{R} = \{(\mathbb{x}, \mathbb{y}, \mathbb{w})\}$. A state function for $(\mathbf{P}, \mathbf{V})$ is a (possibly inefficient) function State that receives as inputs $\mathbb{x}$, $\mathbb{y}$, and a transcript tr and outputs a bit, and has the following properties:

- Empty transcript: if $\mathsf{tr} = \emptyset$ is the empty transcript, then $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 1$ if and only if $(\mathbb{x}, \mathbb{y}) \in L(\mathcal{R})$.
- Prover moves: if tr is a transcript where the prover is about to move, and $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 0$, then for every prover message π, $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}||\pi) = 0$.
- Full transcript: if tr is a full transcript and $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 0$, then $\mathbf{V}$ rejects given this interaction transcript.
- Default: whenever tr is any transcript and $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 1$, then every extension tr' of tr satisfies $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}') = 1$.

With this definition at hand, we now think about a malicious prover that wants to convince the verifier. It starts in state 0 but must manage to end up in state 1, as otherwise the verifier will reject. Thus, in at least one of the rounds, this malicious prover must be lucky in the sense that the state changes from 0 to 1. Round-by-round soundness asserts that changing the state from 0 to 1 happens only with small probability. A proof of proximity relaxes this requirement and allows the probability of changing state to be large when the instance is close to the language, as long as this probability decreases and becomes small when the instance is far from the language.

Definition 3.5 (Round-by-round soundness)**.** A k-round IOPP $(\mathbf{P}, \mathbf{V})$ for a relation $\mathcal{R} = \{(\mathbb{x}, \mathbb{y}, \mathbb{w})\}$ has *round-by-round soundness* with error functions $(\varepsilon_1, \ldots, \varepsilon_{\mathsf{k}})$ if the IOPP has a state function State, such that for every $\mathbb{x}$, $\mathbb{y}$ and transcript $\mathsf{tr} = (\alpha_1, \pi_2, \ldots, \alpha_{i-1}, \pi_i)$,

$$\Pr_{\alpha_i}\left[\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 0 \wedge \mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}||\alpha_i) = 1\right] \leq \varepsilon_i(\mathbb{x}, \mathbb{y}).$$

As with standard soundness, we write ε_i as a function of distance when appropriate.

Clearly, by the above reasoning, round-by-round soundness implies standard soundness by taking a union bound over all errors in the individual rounds. In practice, however, we are mostly interested in the non-interactive BCS-compiled [BCS16] protocol. The soundness of this non-interactive protocol relates directly to round-by-round soundness and not to soundness. To get an intuition for that, note that a malicious prover in the non-interactive setting can try to query the random oracle that outputs the verifier messages multiple times per round. For instance, it may try a few queries to find a good α_1, then proceed to the next round and so on. With soundness, we cannot directly bound the success probability of such a non-interactive prover, as the interactive prover in the soundness experiment only gets one challenge per round. On the other hand, round-by-round soundness can be used, and we briefly and informally sketch the intuition here:

Assume the malicious prover starts with a non-codeword that is far from the code. Assume that it makes Q_i queries to the random oracle for each round i. Then, with every query it can change the state from 0 to 1 with probability at most ε_i, so the probability that it gets lucky at some point is at most

$$Q_1\varepsilon_1 + \cdots + Q_{\mathsf{k}}\varepsilon_{\mathsf{k}} \leq (Q_1 + \cdots + Q_{\mathsf{k}}) \max_i \varepsilon_i.$$

Note that this prover runs in time at least $Q_1 + \cdots + Q_{\mathsf{k}}$. Therefore, the security level of such a scheme is about[3] $\lfloor -\log \max_i \varepsilon_i \rfloor$.

[3]FRI is often used in combination with other building blocks that have their own soundness errors, and the BCS transform itself also introduces additional error terms, e.g., related to the collision-resistance of the hash function in use. So this is only a sketch of how the round-by-round soundness errors relate to the security level.

4 The FRI IOPP

In this section, we recall the FRI IOPP [BBHR18], before we give our new analysis in the next section. We consider the setting in which a prover holds an initial function $f_0 \colon \mathcal{L}_0 \to \mathbb{F}$, to which the verifier has oracle access. Informally, FRI allows the verifier to test that this initial function is close to some initial Reed-Solomon code $\mathcal{C}_0$, of dimension d_0 and rate $\rho = d_0/|\mathcal{L}_0|$. We assume some basic familiarity with the FRI protocol and its associated tools. For a good introduction, we refer to the lectures of Dan Boneh available online [Whi25a, Whi25b].

4.1 Folding univariate functions

We first recall some notation for folding univariate functions following notation of STIR and WHIR [ACFY24a, ACFY24b]. Folding is one of the key tools in the FRI protocol as it allows to reduce the size of a proximity testing problem by a factor m. Definition 4.2 below is given for an arbitrary folding factor m. Readers may be more familiar with the case $m = 2$ where $\widehat{p}_{f,z}$ is computed from the odd and even coefficients of $\widehat{f}$ (in the honest case).

Definition 4.1 (Block)**.** Let $\mathcal{L} \subseteq \mathbb{F}$ be m-smooth. For each $z \in \mathcal{L}^m$, we define the set

$$\mathsf{Block}(\mathcal{L}, m, z) := \{y \in \mathcal{L} \,:\, y^m = z\} \subseteq \mathcal{L}.$$

When $\mathcal{L}$ and m are clear from context, we write $\mathsf{Block}(z)$.

Definition 4.2 (Fold function)**.** Let $\mathcal{L} \subseteq \mathbb{F}$ be m-smooth. Let $f \colon \mathcal{L} \to \mathbb{F}$ be a function and $\alpha \in \mathbb{F}$. For each $z \in \mathcal{L}^m$, define $\widehat{p}_{f,z}$ to be the unique polynomial of degree less than m with $\widehat{p}_{f,z}(y) = f(y)$ for each $y \in \mathsf{Block}(z)$. Then, we define the function $\mathsf{Fold}(f, m, \alpha) \colon \mathcal{L}^m \to \mathbb{F}$ as

$$\forall z \in \mathcal{L}^m,\ \mathsf{Fold}(f, m, \alpha)(z) := \widehat{p}_{f,z}(\alpha).$$

By definition, $\mathsf{Fold}(f, m, \alpha)(z)$ can be computed by interpolation from the evaluations of f at all points in $\mathsf{Block}(z)$.

Remark 4.3. It is well-known (cf. [BS05, ACFY24a, ACFY24b]) that if f is in $\mathsf{RS}[\mathbb{F}, \mathcal{L}, d]$, then $\mathsf{Fold}(f, m, \alpha)$ is in $\mathsf{RS}[\mathbb{F}, \mathcal{L}^m, d/m]$, for every $\alpha \in \mathbb{F}$. This is important for the completeness of FRI.

4.2 Protocol description

Next, we restate the formal protocol description. Correctness follows by inspection.

Construction 4.4 (FRI protocol [BBHR18])**.** Consider the following:

- Initial setting:
  - a field $\mathbb{F}$;
  - an evaluation domain $\mathcal{L}_0 \subseteq \mathbb{F}$ that is a multiplicative coset of $\mathbb{F}^*$;
  - an initial degree bound $d_0 \in \mathbb{N}$ such that $d_0 \leq |\mathcal{L}_0|/2$;
  - an initial code $\mathcal{C}_0 := \mathsf{RS}[\mathbb{F}, \mathcal{L}_0, d_0]$ with rate $\rho := d_0/|\mathcal{L}_0|$;
- Protocol parameters:
  - a number of folding rounds $k \in \mathbb{N}$ and a total number of rounds $\mathsf{k} := k + 1$;
  - a sequence of folding parameters $m_1, \ldots, m_k \in \mathbb{N}$ such that the order of $\mathcal{L}_0$ is divisible by $\prod_{i=1}^{k} m_i$;
  - a query repetition parameter $t \in \mathbb{N}$.

For convenience, we define for every $i \in [k]$:

- the i-th degree bound $d_i := d_{i-1}/m_i$;
- the i-th evaluation domain $\mathcal{L}_i := \mathcal{L}_{i-1}^{m_i} = \{x^{m_i} : x \in \mathcal{L}_{i-1}\}$ of order $|\mathcal{L}_i| = |\mathcal{L}_{i-1}|/m_i$;
- the i-th Reed-Solomon code $\mathcal{C}_i := \mathsf{RS}[\mathbb{F}, \mathcal{L}_i, d_i]$.

Note that by definition, all codes $\mathcal{C}_0, \ldots, \mathcal{C}_k$ have the same rate ρ and the order of the $(i-1)$-th domain $|\mathcal{L}_{i-1}|$ is divisible by $\prod_{j=i+1}^{k} m_j$. In particular, $\mathcal{L}_{i-1}$ is m_i-smooth. Finally, given an entry $z \in \mathcal{L}_0$ and round number $i \in \{0, \ldots, k\}$, we define the following shorthand notation consistent with the definitions of the "local fold check" below:

$$z^{(i)} := \begin{cases} z, & i = 0, \\ \left(z^{(i-1)}\right)^{m_i}, & i \geq 1. \end{cases} \tag{1}$$

Protocol description. The FRI protocol proceeds as follows:

- **Initial function.** Let $f_0 \colon \mathcal{L}_0 \to \mathbb{F}$ be an initial function. In the honest case, $f_0 \in \mathcal{C}_0$.
- **Folding phase.** For every round $i = 1, \ldots, k$:
  1. **Folding challenge.** The verifier sends a challenges $\alpha_i \leftarrow \mathbb{F}$.
  2. **Send folded function.** The prover sends a function $f_i \colon \mathcal{L}_i \to \mathbb{F}$. In the honest case, $f_i = \mathsf{Fold}(f_{i-1}, m_i, \alpha_i)$.
- **Query phase.** For every repetition $j = 1, \ldots, t$, the verifier samples $s_j \leftarrow \mathcal{L}_0$. The verifier sends $\alpha_{k+1} := (s_1, \ldots, s_t)$.
- **Verifier decision phase:**
  1. **Final function check.** The verifier reads the final function f_k in full and asserts that $f_k \in \mathcal{C}_k$.
  2. **Local fold checks.** For every repetition $j = 1, \ldots, t$:
     - Set $s_j^{(0)} = s_j$.
     - For every folding round $i = 1, \ldots, k$:
       * Set $s_j^{(i)} := (s_j^{(i-1)})^{m_i}$.
       * Query $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(s_j^{(i)})$. This involves querying f_{i-1} at all the points $x \in \mathsf{Block}(s_j^{(i)})$.
       * Assert that $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(s_j^{(i)}) = f_i(s_j^{(i)})$.

Choosing parameters. In practice, we often choose $\mathbb{F}$ such that the order of $\mathbb{F}^*$ is divisible by a large power of 2 and pick the values $m_1, \ldots, m_k$ to be powers of 2.

5 Our Proof

In this section, we present our new security analysis. We start with an informal overview in Section 5.1. We then state and prove a key lemma about folding univariate functions in Section 5.2. Using this lemma, we give our formal proof in Section 5.3. Finally, we give a high-level comparison of our proof with exists strategies in Section 5.4.

5.1 Intuition and overview

To prove that FRI has round-by-round soundness, we first define a state function. Then, for each round of interaction, we will bound the probability that a random verifier message changes the state from 0 to 1. Before describing the state function, we introduce the *graph of prover messages*, a useful tool in our reasoning which appears implicitly [BCI+20] or explicitly [ABN23] in previous analyses of FRI.

The prover message graph. Consider an execution of the FRI protocol for the initial function f_0 with k rounds of folding. A full transcript for this interaction is of the form

$$\mathsf{tr} = (\alpha_1, f_1, \ldots, \alpha_k, f_k, \alpha_{k+1}),$$

where the values $\alpha_1, \ldots, \alpha_k$ and $f_1, \ldots, f_k$ are respectively the verifier and prover messages during the folding rounds, and α_{k+1} defines the verifier's queries to the oracles. By definition of the FRI protocol, the functions $f_0, \ldots, f_k$ are defined over related domains ($\mathcal{L}_0, \ldots, \mathcal{L}_k$, respectively) and are expected to be consecutive applications of the Fold function to f_0.

We can draw a layered graph in which the i-th layer represents the i-th evaluation domain $\mathcal{L}_i$, and each node of the layer corresponds to a point $y \in \mathcal{L}_i$. We draw an edge between nodes $x \in \mathcal{L}_{i-1}$ and $y \in \mathcal{L}_i$ if and only if $x^{m_i} = y$, or equivalently $x \in \mathsf{Block}(y)$. In this case, we follow typical graph conventions and call y the *parent* node and x the *child* node; children nodes with the same parent node are called *siblings*. An illustration of this graph for $|\mathcal{L}_0| = 16$, $k = 3$ and $m_i = 2$ for all i is given in[4] Figure 1.

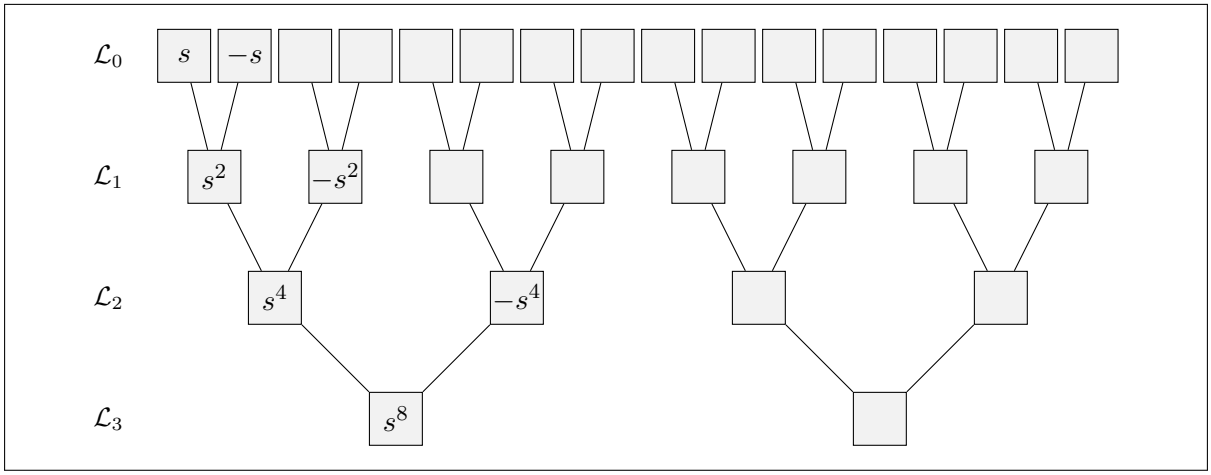

Figure 1: A prover message graph before coloring. We label some values to make explicit the relation between parent and children nodes.

[4]We sincerely apologize to all computer scientists whose hearts ache at the sight of two upside-down trees — we draw them this way to show the temporal order of prover messages.

Coloring the graph. We will now *color* the graph to reflect the verifier's observations and its eventual output. During the query phase of FRI, the verifier reads the final function f_k in full and runs tests to assert that the rest of the graph is consistent with it. The verifier samples a node $s^{(0)} \in \mathcal{L}_0$ and defines a path of related queries $s^{(1)} \in \mathcal{L}_1, \ldots, s^{(k)} \in \mathcal{L}_k$ that traverses the graph vertically. Roughly, our coloring therefore considers all such paths and marks them as red if any of the verifier's assertions fail, and green otherwise. More concretely, the coloring rules are defined below and illustrated in Figure 2:

1. *Base case.* The bottom layer is always green.
2. *Inductive rule.* Given a node $x \in \mathcal{L}_{i-1}$ at layer $i-1$ with parent $y \in \mathcal{L}_i$:
   (a) if the parent y is red, then x is red. This reflects the fact that any verification path that goes through x will also go through $x^{m_i} = y$.
   (b) if the parent y is green, compute $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(y)$ from x and its siblings and
      i. color x green if an only if $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(y) = f_i(y)$ (*i.e.*, if the verifier's assertion passes).
      ii. color x red otherwise (*i.e.*, if the verifier's assertion fails).

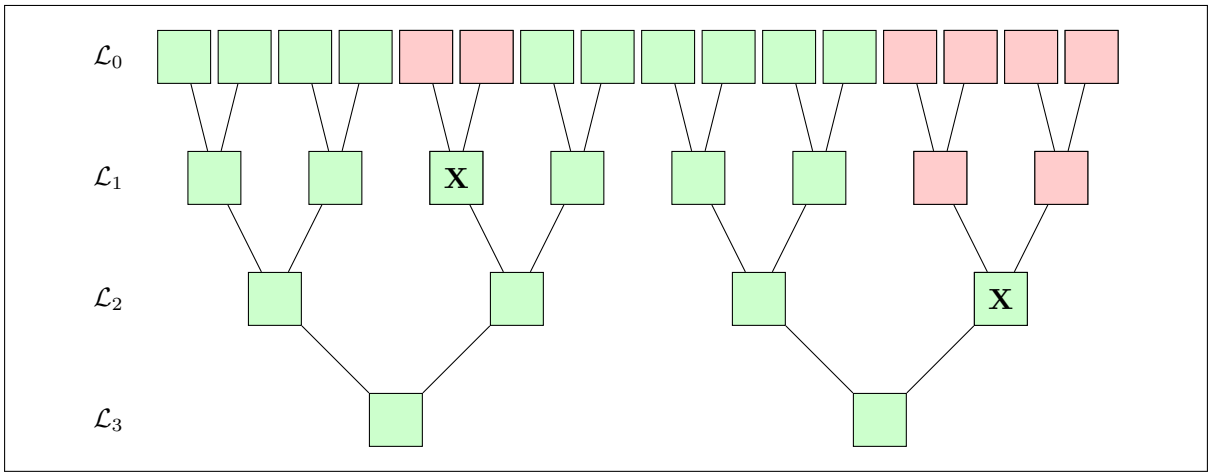

Figure 2: A colored prover message graph. We mark a node $y \in \mathcal{L}_i$ with an **X** if the value $f_i(y)$ is not consistent with $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(y)$.

We stress the following important intuition about the prover message graph:

- if the verifier sample $s^{(0)}$ is at a red node in the top layer, the verifier will reject.
- red propagates upwards: if the prover folds incorrectly at any layer, the consequences are felt all the way to the top layer.
- the green nodes in the graph correspond to subsets W_i of the evaluation domains $\mathcal{L}_i$ where the Fold function was applied honestly, that is $f_i[W_i] = \mathsf{Fold}(f_{i-1}, m_i, \alpha)[W_i]$.

Given these observations, we are ready to informally describe and analyze our state function.

State function. By definition, we need to set the initial state $\mathsf{State}(f_0, \emptyset)$ to be 0 if and only if $f_0 \notin \mathcal{C}_0$. We now separate the description of the state function between the folding rounds and the query round.

- *Folding rounds.* As noted in our intuition for the prover message graph, the query phase is designed such that the verifier will reject queries that fall outside of the green subsets. Therefore, during the folding phase we can focus on the green subsets and reason only in terms of honestly folded functions; trusting that the red subgraph will be rejected in the query phase.

  Ideally, we would like to establish that if $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)$ agrees with a codeword $u_i \in \mathcal{C}_i$ on the green subset $W_i \subseteq \mathcal{L}_i$, then f_{i-1} agrees with a codeword $u_{i-1} \in \mathcal{L}_{i-1}$ on the green subset $W_{i-1} \subseteq \mathcal{L}_{i-1}$. In other words, that:

  > *(ideal property)* "agreement propagates upwards through green nodes."

  Unfortunately, this property does not always hold true. We can, however, show that a relaxed version of this implication holds with high probability using mutual correlated agreement. Our relaxed ideal property becomes

  > *(relaxed ideal property)* "if all green subsets are *large enough*, then agreement propagates upwards through green nodes."

  We will define what *large enough* means very shortly. We can accept such a relaxation of the ideal property because if all green subsets are not large, then the red subsets must be large and the verifier will reject with high probability. We will discuss this further in the analysis section.

  We define the state function such that it indicates whether the relaxed ideal property holds for a given initial function f_0 and transcript tr: when all the honest folded functions are safe (as per our relaxed ideal property), we set $\mathsf{State}(f_0, \mathsf{tr}) = 0$; when there is an honest fold that is exploitable (one for which all green subsets are large but agreement does not propagate upward), we set $\mathsf{State}(f_0, \mathsf{tr}) = 1$. The event "the state changed from 0 to 1" is captured by the following formal description. Given an initial function f_0 and a transcript $\mathsf{tr} := (\alpha_1, f_1, \ldots, \alpha_{i-1}, f_{i-1})$ such that $\mathsf{State}(f_0, \mathsf{tr}) = 0$, and an i-th verifier message α_i, we say that $\mathsf{State}(f_0, \mathsf{tr}||\alpha_i) = 1$ if and only if there exists a set $S \subseteq \mathcal{L}_{i-1}$ such that:

  1. The green subsets are "large enough" at layers $i-1$ and i. Formally, $|S^{m_i}| \geq (1-\delta^*) \cdot |\mathcal{L}_i|$, where $(1-\delta^*)$ is the notion of "large enough" that we will specify later. Note that this implies that $|S| \geq (1-\delta^*) \cdot \mathcal{L}_{i-1}$.
  2. The honest fold of f_{i-1} agrees with a codeword $u \in \mathcal{C}_i$ on the set S^{m_i}.
  3. There are no codewords in $\mathcal{C}_{i-1}$ that agree with f_{i-1} on the set S.

- *Query round.* The query round is the final round of the protocol. Therefore, by the "full transcript" property of the state function (see Definition 3.4), the output of the state function at this round must be equivalent to the verifier's final decision. We therefore set the state function to be 1 if and only if all the verifier checks pass.

Analysis. To complete our proof of round-by-round soundness, we must bound the probability at round i that the state changes from 0 of 1. We separate the analysis between the folding rounds and the query round.

- *Folding rounds.* The analysis of the folding rounds relies on the mutual correlated agreement property. This can be applied because the Fold function is indeed a linear combination that uses the Powers proximity generator. Any challenge α_i that causes an exploitable fold at round i (as defined by the state function) is also a challenge that realizes the bad event of

mutual correlated agreement. This intuition is formally stated and proven in Lemma 5.1. Letting $\mathsf{B}^\star$ and $\mathsf{err}^\star$ be the bound and error associated with the Powers proximity generator, we establish that

$$\varepsilon_i^{\mathsf{fold}}(\delta) := \mathsf{err}^\star(\mathcal{C}_i, m_i, \delta^*),$$

where $\delta := \Delta(f_0, \mathcal{C}_0)$ and $\delta^* := \min\{\delta, 1 - \mathsf{B}^\star(\rho)\}$. The latter value is set to satisfy the premise of mutual correlated agreement (and in turn of Lemma 5.1). In doing so, we finally define the notion of a "large enough" green subset.

- *Query round.* The analysis of the query round follows directly from our discussion on the prover message graph. Consider an initial function f_0 and a transcript $\mathsf{tr} := (\alpha_1, f_1, \ldots, \alpha_k, f_k)$ such that $\mathsf{State}(f_0, \mathsf{tr}) = 0$, and a final verifier message α_{k+1}. Notice that if $f_k \notin \mathcal{C}_k$, then the verifier rejects immediately. Therefore, we will assume that $f_k \in \mathcal{C}_k$.

  From the definition of our state function, we know that $\mathsf{State}(f, \mathsf{tr}||\alpha_{k+1}) = 1$ if and only if the verifier's sample is in the green subset $W_0 \subseteq \mathcal{L}_0$. In other words, the probability that the state changes from a single query is exactly $|W_0|/|\mathcal{L}_0|$. We can now separate our analysis in two mutually exclusive cases:

  1. if any green subsets is small, then $|W_0|/|\mathcal{L}_0| \leq (1 - \delta^*)$. This follows from the fact that red propagates upwards when coloring the prover message graph.
  2. if all green subsets are large, then we know that "agreement propagates upwards through green nodes". This follows from the fact that $\mathsf{State}(f, \mathsf{tr}) = 0$ and our definition of the state function for folding rounds. Therefore, there exists a codeword $u_0 \in \mathcal{C}_0$ such that $f_0[W_0] = u_0[W_0]$. Consequently

     $$\frac{|W_0|}{|\mathcal{L}_0|} = 1 - \Delta(f_0, u_0) \leq 1 - \Delta(f_0, \mathcal{C}_0) \leq 1 - \delta^*,$$

     also establishing the desired upper bound.

  The upper bound holds in both cases. To complete the proof, we simply note that all t repetitions of the query phase are independent. Therefore, we establish that

  $$\varepsilon^{\mathsf{qry}} := (1 - \delta^*)^t.$$

  We give all formal details of our proof in the following sections.

5.2 Folding preserves agreement sets

Here we state and prove a simple lemma that is crucial to our analysis. Intuitively, we consider folding a function with a random folding challenge r, and show that the distance to the code is does not go down. The lemma states even more: except with small probability over r, for all large enough sets S^m on which the folded function agrees with the code, the original function agrees on S with its code. We show that this follows from mutual correlated agreement. The proof of the lemma is a generalization of Lemma 4.9 in STIR [ACFY24a].

Lemma 5.1 (Folding preserves agreement set)**.** *Let $m \in \mathbb{N}$ be a folding factor, $\mathcal{L} \subseteq \mathbb{F}$ be an m-smooth evaluation domain and $d \in \mathbb{N}$ be a degree bound with $d \geq m$. Consider the Reed-Solomon codes $\mathcal{C}_0 := \mathsf{RS}[\mathbb{F}, \mathcal{L}, d]$ and $\mathcal{C}_1 := \mathsf{RS}[\mathbb{F}, \mathcal{L}^m, d/m]$, both with rate $\rho = d/|\mathcal{L}|$.*

If Powers *is a proximity generator with mutual correlated agreement (cf. Definition 2.2) with proximity bound $\mathsf{B}^\star$ and error $\mathsf{err}^\star$, then for every function $f \colon \mathcal{L} \to \mathbb{F}$ and distance parameter*

$\delta \in [0, 1 - \mathsf{B}^\star(\rho))$,

$$\Pr_{r \leftarrow \mathbb{F}} \left[\exists S \subseteq \mathcal{L} \; s.t. \; \begin{array}{l} |S^m| \geq (1-\delta) \cdot |\mathcal{L}^m| \\ \wedge \exists u' \in \mathcal{C}_1, \mathsf{Fold}(f, m, r)[S^m] = u'[S^m] \\ \wedge \forall u \in \mathcal{C}_0, u[S] \neq f[S] \end{array} \right] \leq \mathsf{err}^\star(\mathcal{C}_1, m, \delta).$$

Proof. Let $f \colon \mathcal{L} \to \mathbb{F}$ be a function, and define functions $f_1, \ldots, f_m \colon \mathcal{L}^m \to \mathbb{F}$ so that

$$\mathsf{Fold}(f, m, r) = \sum_{i=1}^{m} r^{i-1} f_i \; : \mathcal{L}^m \to \mathbb{F}, \tag{2}$$

as follows. Recall the polynomials $\widehat{p}_{f,z}$ as defined in Definition 4.2, *i.e.*, for every $z \in \mathcal{L}^m$, $\widehat{p}_{f,z}$ is the unique polynomial of degree less than m that agrees with f on $\mathsf{Block}(z)$. We let $f_i \colon \mathcal{L}^m \to \mathbb{F}$ be the function that maps z to the i-th coefficient of $\widehat{p}_{f,z}$, *i.e.*, so that $\widehat{p}_{f,z}(Y) = \sum_{i=1}^{m} f_i(z) Y^{i-1}$ for every $z \in \mathcal{L}^m$. One can see that these functions f_i satisfy (2), by definition of Fold (cf. Definition 4.2).

Now, using the equality (2) to substitute $\mathsf{Fold}(f, m, r)$ in the probability we want to bound, we have

$$\begin{aligned} &\Pr_{r} \left[\exists S \subseteq \mathcal{L} \; s.t. \; \begin{array}{l} |S^m| \geq (1-\delta_0) \cdot |\mathcal{L}^m| \\ \wedge \exists u' \in \mathcal{C}_1, \mathsf{Fold}(f, m, r)[S^m] = u'[S^m] \\ \wedge \forall u \in \mathcal{C}_0, u[S] \neq f[S] \end{array} \right] \\ \leq &\Pr_{r} \left[\exists S \subseteq \mathcal{L} \; s.t. \; \begin{array}{l} |S^m| \geq (1-\delta_0) \cdot |\mathcal{L}^m| \\ \wedge \exists u' \in \mathcal{C}_1, \sum_{i=1}^{m} r^{i-1} f_i[S^m] = u'[S^m] \\ \wedge \forall u \in \mathcal{C}_0, u[S] \neq f[S] \end{array} \right]. \end{aligned}$$

The latter is almost the event whose probability is bounded in the definition of mutual correlated agreement (cf. Definition 2.2) for the code $\mathcal{C}_1 = \mathsf{RS}[\mathbb{F}, \mathcal{L}^m, d/m]$, except for the last condition, which reads

$$\forall u \in \mathcal{C}_0, u[S] \neq f[S], \tag{3}$$

while in the aforementioned event, the corresponding condition is

$$\exists i \in [m], \forall u' \in \mathcal{C}_1, u'[S^m] \neq f_i[S^m]. \tag{4}$$

Thus, to prove the present lemma, it suffices to prove that (3) implies (4).

To this end, assume towards contradiction that (4) does not hold, *i.e.,* for every $i \in [m]$, there is a $u_i \in \mathcal{C}_1$ such that f_i and u_i agree on S^m. We will show that then f agrees with a codeword in $\mathcal{C}_0$ on S. To do so, we will first define a bivariate polynomial $\widehat{Q}(X, Y) \in \mathbb{F}[X, Y]$ and will show that f agrees with $\widehat{Q}(X^m, X)$ on S. We will also show that $\widehat{Q}(X^m, X)$ has degree less than d, which implies our desired result, namely that f agrees on S with a codeword from $\mathcal{C}_0$.

Precisely, we define the bivariate polynomial $\widehat{Q} \in \mathbb{F}[X, Y]$ as follows: Fix any subset $\tilde{S} \subseteq S^m$ of size $\min\{|S^m|, d/m\}$. For every $x \in \tilde{S}$, let $I_x \in \mathbb{F}[X]$ denote the polynomial of degree less than $|\tilde{S}|$ such that

$$I_x(z) = \begin{cases} 1 & \text{if } z = x, \\ 0 & \text{if } z \in \tilde{S} \setminus \{x\} \end{cases}.$$

Note that the degree of I_x is less than d/m since $|\tilde{S}| \leq d/m$. Now, define

$$\widehat{Q}(X, Y) := \sum_{x \in \tilde{S}} I_x(X) \cdot \widehat{p}_{f,x}(Y) \in \mathbb{F}[X, Y].$$

Clearly, $\widehat{Q}(X,Y)$ has degree less than d/m on the variable X, and degree less than m on the variable Y.

Next, we show that for every $x \in S^m$, we have $\widehat{Q}(x,Y) = \widehat{p}_{f,x}(Y)$ as polynomials. Indeed, we have

$$\widehat{Q}(\tilde{x},Y) = \sum_{x\in\tilde{S}} I_x(\tilde{x}) \cdot \widehat{p}_{f,x}(Y) = \widehat{p}_{f,\tilde{x}}(Y) = \sum_{i=1}^{m} f_i(\tilde{x})Y^{i-1} = \sum_{i=1}^{m} u_i(\tilde{x})Y^{i-1} \quad \text{for all } \tilde{x} \in \tilde{S}, \quad (5)$$

where the first equality is by definition of $\widehat{Q}$; the second equality follows from the definition of the polynomials $I_x(X)$; the third from the definition of the functions f_i; and the fourth from the assumption that f_i agrees with the codeword u_i on $S^m \supseteq \tilde{S}$, for all $i \in [m]$.

Assume now that $|\tilde{S}| = d/m$, and recall that $\widehat{Q}(X,Y)$ has degree less than d/m on the variable X. In particular, $\widehat{Q}(X,Y)$ can be seen as a univariate polynomial on the variable X, of degree less than d/m, with coefficients on the field of rational functions $\mathbb{F}(Y)$. Similarly, $\sum_{i=1}^{m} u_i(X)Y^{i-1}$ can be seen as a univariate polynomial on the variable X of degree less than d/m, with coefficients in $\mathbb{F}(Y)$. Since we assumed $|\tilde{S}| = d/m$, because of (5), these two polynomials agree on d/m distinct points, and so they must be equal as polynomials. Hence, in this case,

$$\widehat{Q}(X,Y) = \sum_{i=1}^{m} u_i(X)Y^{i-1},$$

and in particular

$$\widehat{Q}(x,Y) = \sum_{i=1}^{m} u_i(x)Y^{i-1} = \widehat{p}_{f,x}(Y) \quad \text{for all } x \in S^m, \quad (6)$$

as claimed. On the other hand, if $|\tilde{S}| < d/m$, then $\tilde{S} = S^m$, and then (6) holds for all $x \in S^m$ because of Equation (5).

Finally, define the polynomial $\widehat{f}(X) := \widehat{Q}(X^m, X)$, which has degree less than d because $\widehat{Q}(X^m, Y)$ has degree on X at most $m(d/m-1)$, and so $\widehat{Q}(X^m, X)$ has degree on X at most

$$m(d/m-1) + (m-1) = d - m + m - 1 = d - 1.$$

Now, note that $\widehat{f}$ and f agree on S, because for every $x \in S$:

$$\widehat{f}(x) = \widehat{Q}(x^m, x) = \widehat{p}_{f,x^m}(x) = f(x),$$

where the first equality follows by definition of $\widehat{f}$, the second from Equation (6), and the third by definition of $\widehat{p}_{f,x^m}$, since $\widehat{p}_{f,x^m}$ is unique polynomial of degree less than m agreeing with f on $\mathsf{Block}(x^m)$, and $x \in \mathsf{Block}(x^m)$. Hence, f agrees with the codeword $\widehat{f} \in \mathsf{RS}[\mathbb{F}, \mathcal{L}, d]$ on S, as needed. This completes the proofs that (3) implies (4), and therefore completes the proof of the desired lemma. □

5.3 Full proof

We now state the theorem for round-by-round soundness and give its full proof.

Theorem 5.2. *Consider FRI with parameters as in Construction 4.4. If* Powers *is a proximity generator with mutual correlated agreement (cf. Definition 2.2) with proximity bound* $\mathsf{B}^\star$ *and error* $\mathsf{err}^\star$*, then FRI has round-by-round soundness with error functions* $\varepsilon_1^{\mathsf{fold}}, \ldots, \varepsilon_k^{\mathsf{fold}}, \varepsilon^{\mathsf{qry}} \colon (0,1] \to [0,1]$ *such that for every initial function* $f_0 \notin \mathcal{C}_0$*:*

- *for all* $i \in [k]$, $\varepsilon_i^{\mathsf{fold}}(\delta) = \mathsf{err}^\star(\mathcal{C}_i, m_i, \delta^*)$,
- $\varepsilon^{\mathsf{qry}}(\delta) = (1-\delta^*)^t$,

where $\delta := \Delta(f_0, \mathcal{C}_0)$ *and* $\delta^* := \min\{\delta, 1 - \mathsf{B}^\star(\rho) - 1/|\mathcal{L}_0|\}$.

Proof. We describe the state function State and for each round give an upper bound on the probability that the state changes from 0 to 1. By default, the state function never changes from 1 back to 0, and the state function does not change if the prover moves. We describe below how the state function may change from 0 to 1 when the verifier moves, and bound the probability of such a change happening.

0. **Empty transcript.** For an empty transcript, we say that $\mathsf{State}(f_0, \emptyset) = 1$ if and only if $f_0 \in \mathcal{C}_0$.

1. **Folding rounds** $(\varepsilon_1^{\mathsf{fold}}, \ldots, \varepsilon_k^{\mathsf{fold}})$**.** For rounds $i \in \{1, \ldots, k\}$, the transcript tr has the form

$$\mathsf{tr} = \begin{cases} \emptyset, & i = 1, \\ (\alpha_1, f_1, \ldots, \alpha_{i-1}, f_{i-1}), & 1 < i \leq k, \end{cases}$$

and the verifier sends a random message $\alpha_i \in \mathbb{F}$.

- *State function.* Let $\delta^* := \min\{\Delta(f_0, \mathcal{C}_0), 1-\mathsf{B}^\star(\rho)-1/|\mathcal{L}_0|\}$. We say that $\mathsf{State}(f_0, \mathsf{tr}||\alpha_i) = 1$ if there exists a set $S \subseteq \mathcal{L}_{i-1}$ such that:
  (a) $|S^{m_i}| \geq (1-\delta^*) \cdot |\mathcal{L}_i|$; and
  (b) there exists $u' \in \mathcal{C}_i$ such that $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)[S^{m_i}] = u'[S^{m_i}]$; and
  (c) for all $u \in \mathcal{C}_{i-1}$, there exists $x \in S$ such that $u(x) \neq f_{i-1}(x)$.
- *Bounding the error.* Fix $i \in [k]$ and assume that $\mathsf{State}(f_0, \mathsf{tr}) = 0$. Notice that $\delta^* \in [0, 1-\mathsf{B}^\star)$ by definition and that the event that causes a state change is exactly the one described in Lemma 5.1 for proximity parameter δ^*. Therefore, applying the lemma to the function f_{i-1}, we establish that:

$$\Pr_{\alpha_i}[\mathsf{State}(f_0, \mathsf{tr}) = 0 \ \wedge \ \mathsf{State}(f_0, \mathsf{tr}||\alpha_i) = 1] \leq \mathsf{err}^\star(\mathcal{C}_i, m_i, \delta^*).$$

  Setting $\varepsilon_i^{\mathsf{fold}}(\delta) := \mathsf{err}^\star(\mathcal{C}_i, m_i, \delta^*)$ completes the proof for all folding rounds.

2. **Query round** $\varepsilon^{\mathsf{qry}}$**.** A final round partial transcript tr has the form $\mathsf{tr} = (\alpha_1, f_1, \ldots, \alpha_k, f_k)$. The final verifier message α_{k+1} has the form $\alpha_{k+1} = (s_1, \ldots, s_t)$ with $s_1, \ldots, s_t \in \mathcal{L}_0$.

- *State function.* We say that $\mathsf{State}(f_0, \mathsf{tr}||\alpha_{k+1}) = 1$ if both of the following conditions are satisfied:
  (a) the final prover message f_k is in the corresponding code $\mathcal{C}_k$, and
  (b) for all $j \in [t]$, for all $i \in [k]$, $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(s_j^{(i)}) = f_i(s_j^{(i)})$, where $s_j^{(i)}$ is defined as in the verifier decision phase (and Equation (1)).
- *Bounding the error.* Suppose that $\mathsf{State}(f_0, \mathsf{tr}) = 0$. We can assume that Item 2a (final prover message is in the code) is true, otherwise the state will remain 0. Throughout this section, we will use the notation $z^{(i)}$ as defined in Equation (1).

  Defining $\mathsf{AccQueries}$**.** Let $\mathsf{AccQueries} \subseteq \mathcal{L}_0$ denote the (potentially empty) set of queries $z \in \mathcal{L}_0$ such that for every round index $i \in [k]$, $\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(z^{(i)}) = f_i(z^{(i)})$. In other words, $\mathsf{AccQueries}$ is the set of positions in $\mathcal{L}_0$ for which the verifier's "local fold checks" are satisfied for every prover message.

By definition of $\mathsf{AccQueries}$ and the state function, it holds that $\mathsf{State}(f_0, \mathsf{tr}||\alpha_{k+1}) = 1$ if and only if for every repetition $j \in [t]$ of the query phase, $s_j \in \mathsf{AccQueries}$. Therefore, it holds that:

$$\begin{aligned}\Pr_{\alpha_{k+1}}\left[\begin{array}{c}\mathsf{State}(f_0,\mathsf{tr})=0\\ \wedge\\ \mathsf{State}(f_0,\mathsf{tr}||\alpha_{k+1})=1\end{array}\right] &= \Pr_{s_1,\ldots,s_t\leftarrow\mathcal{L}_0}[\forall j\in[t], s_j\in\mathsf{AccQueries}]\\ &= \left(\frac{|\mathsf{AccQueries}|}{|\mathcal{L}_0|}\right)^t.\end{aligned} \tag{7}$$

Upper bounding $|\mathsf{AccQueries}|$**.** Let $\delta^* := \min\{\Delta(f_0, \mathcal{C}_0), 1 - \mathsf{B}^\star(\rho) - 1/|\mathcal{L}_0|\}$. The rest of the proof aims to show that

$$|\mathsf{AccQueries}|/|\mathcal{L}_0| \leq 1 - \delta^*.$$

We start by defining the sets $W_0, \ldots, W_k$ inductively as follows:

- base case: $W_0 := \mathsf{AccQueries}$,
- induction: $W_{i+1} := \{x^{m_{i+1}} : x \in W_i\} = W_i^{m_{i+1}}$,

and use Claims 5.3 and 5.4 to prove the desired inequality. For readers familiar with Section 5.1, we will prove later that the sets $W_0, \ldots W_k$ are subsets of $\mathcal{L}_0, \ldots, \mathcal{L}_k$ for which all the nodes are green and all their immediate children are green (*i.e.*, nodes that are green and not marked with an **X** in Figure 2).

Claim 5.3. *If there exists $i^* \in \{0, \ldots, k\}$ such that $|W_{i^*}| < (1 - \delta^*) \cdot |\mathcal{L}_{i^*}|$, then $|W_0| < (1 - \delta^*) \cdot |\mathcal{L}_0|$.*

Proof. First notice that for $i^* = 0$, the statement holds trivially. We now prove it for $i^* \in [k]$. Note also that W_{i-1} is contained in the set $U = \bigcup_{x \in W_i} \mathsf{Block}(x)$. Then, since the sets $\mathsf{Block}(x)$ are disjoint, we have $|U| = m_i \cdot |W_i|$, and so

$$|W_{i-1}| \leq m_i \cdot |W_i|.$$

Recalling that $|\mathcal{L}_{i-1}| = m_i \cdot |\mathcal{L}_i|$, it holds that

$$\frac{|W_{i-1}|}{|\mathcal{L}_{i-1}|} \leq \frac{|W_i|}{|\mathcal{L}_i|}.$$

Therefore, if there exists $i^* \in [k]$ such that $|W_{i^*}| < (1 - \delta^*) \cdot |\mathcal{L}_{i^*}|$, then

$$\frac{|W_0|}{|\mathcal{L}_0|} \leq \frac{|W_{i^*}|}{|\mathcal{L}_{i^*}|} < 1 - \delta^*,$$

proving the claim. □

Claim 5.4. *If $f_k \in \mathcal{C}_k$ and for all $i \in \{0, \ldots, k\}$, $|W_i| \geq (1 - \delta^*) \cdot |\mathcal{L}_i|$, then there exists $u_0 \in \mathcal{C}_0$ such that $u_0[W_0] = f_0[W_0]$.*

Proof. By definition of the sets $W_0, \ldots, W_k$, it holds that for all $i \in \{0, \ldots, k\}$, $W_i = \{z^{(i)} : z \in W_0\}$. We claim that

$$\forall i \in [k],\ f_i[W_i] = \mathsf{Fold}(f_{i-1}, m_i, \alpha_i)[W_i]. \tag{8}$$

For each $i \in [k]$, let A_i be the subset of elements a from $\mathcal{L}_i$ such that $f_i(a) = \mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(a)$. To prove Equation (8), it suffices to show that $W_i \subseteq A_i$. To this end, fix any $i \in [k]$, and let

$w \in W_i$ be arbitrary. As noted above, we can write $w = z^{(i)}$ for some $z \in W_0 = \mathsf{AccQueries}$. By definition of the set $\mathsf{AccQueries}$, we have

$$\mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(w) = \mathsf{Fold}(f_{i-1}, m_i, \alpha_i)(z^{(i)}) = f_i(z^{(i)}) = f_i(w),$$

where we have used the definition of $\mathsf{AccQueries}$ in the second equality. Therefore, $w \in A_i$ by definition of A_i, and thus $W_i \subseteq A_i$ as w was arbitrary. This shows that the claimed equation (Equation (8)) is true.

Next, we claim that for all $i \in \{1, \ldots, k\}$, if there exists $u_i \in \mathcal{C}_i$ such that $u_i[W_i] = f_i[W_i]$ then there exists $u_{i-1} \in \mathcal{C}_{i-1}$, $u_{i-1}[W_{i-1}] = f_{i-1}[W_{i-1}]$. Given that $W_k \subseteq \mathcal{L}_k$ and that $f_k \in \mathcal{C}_k$, we can apply this implication for all $i \in [k]$ and conclude that there exists $u_0 \in \mathcal{C}_0$ such that $u_0[W_0] = f_0[W_0]$.

It remains to prove this claimed implication. Assume for contradiction that there exists $i^* \in \{1, \ldots, k\}$ such that both of the following conditions are satisfied:

- there exists $u_{i^*} \in \mathcal{C}_{i^*}$ such that $u_{i^*}[W_{i^*}] = f_{i^*}[W_{i^*}] = \mathsf{Fold}(f_{i^*-1}, m_{i^*}, \alpha_{i^*})[W_{i^*}]$; and
- for all $u \in \mathcal{C}_{i^*-1}$, $u[W_{i^*-1}] \neq f_{i^*-1}[W_{i^*-1}]$.

Given that $W_{i^*} = (W_{i^*-1})^{m_i^*}$ by definition, it holds that the two bullets points above imply that Items 1b and 1c are satisfied for the set W_{i^*-1}. Since $\mathsf{State}(f_0, \mathsf{tr}) = 0$, this implies that $|W_{i^*}| < (1-\delta^*) \cdot |\mathcal{L}_{i^*}|$ (Item 1a is false). However, the premise of Claim 5.4 states that for all $i \in \{0, \ldots, k\}$, $|W_i| \geq (1-\delta^*) \cdot |\mathcal{L}_i|$. This is a contradiction. So, we have proven the implication, and thus the claim. □

Given Claims 5.3 and 5.4 above, we can separate our analysis into two mutually exclusive cases:

- Case 1: $\exists i \in \{0, \ldots, k\}, |W_i| < (1-\delta^*) \cdot |\mathcal{L}_i|$. Here, it follows from Claim 5.3 that

$$\frac{|\mathsf{AccQueries}|}{|\mathcal{L}_0|} = \frac{|W_0|}{|\mathcal{L}_0|} < \frac{(1-\delta^*)|\mathcal{L}_0|}{|\mathcal{L}_0|} = 1 - \delta^*.$$

- Case 2: $\forall i \in \{0, \ldots, k\}, |W_i| \geq (1-\delta^*) \cdot |\mathcal{L}_i|$. Here, it follows from Claim 5.4 that there exist $u_0 \in \mathcal{C}_0$ such that $u_0[W_0] = f_0[W_0]$. Therefore, by definition of the relative distance to a code:

$$\Delta(f_0, \mathcal{C}_0) \leq \Delta(f_0, u_0).$$

  Noting that $\delta^* \leq \Delta(f_0, \mathcal{C}_0)$ by definition of δ^*, we have that:

$$\delta^* \leq \Delta(f_0, u_0);$$

  or equivalently $1 - \delta^* \geq 1 - \Delta(f_0, u_0)$. Finally, noting that the agreement set between f_0 and u_0 is at least of size $|W_0|$, it holds that $1 - \Delta(f_0, u_0) \geq |W_0|/|\mathcal{L}_0|$. Combining with the above, we find that

$$1 - \delta^* \geq 1 - \Delta(f_0, u_0) \geq \frac{|W_0|}{|\mathcal{L}_0|} = \frac{|\mathsf{AccQueries}|}{|\mathcal{L}_0|},$$

  proving the desired inequality.

In both cases, we have shown that $|\mathsf{AccQueries}|/|\mathcal{L}_0| \leq 1 - \delta^*$. Plugging this into Equation (7), we find that:

$$\Pr_{\alpha_{k+1}}\left[\begin{array}{c} \mathsf{State}(f_0, \mathsf{tr}) = 0 \\ \wedge \\ \mathsf{State}(f_0, \mathsf{tr} || \alpha_{k+1}) = 1 \end{array}\right] \leq (1-\delta^*)^t.$$

Setting $\varepsilon^{\mathsf{qry}}(\delta) := (1-\delta^*)^t$ completes the proof.

3. **Verifier decision.** A full transcript is of the form $\mathsf{tr} = (\alpha_1, f_1, \ldots, \alpha_k, f_k, \alpha_{k+1})$. If we have $\mathsf{State}(f_0, \mathsf{tr}) = 0$, then either Item 2a or Item 2b is false. These conditions are identical to the checks performed by the verifier ("final function check" and "local fold check" respectively.) Therefore, $\mathbf{V}^{f_0, f_1, \ldots, f_k}(\alpha_1, \ldots, \alpha_{k+1}) = 0$ as desired.

□

5.4 Comparison to existing proof strategies

Let us put our proof into context by comparing it with other analyses.

Proofs using CA. Our proof follows the ideas of Ben-Sasson *et al.* [BCI+20] and Augot, Bordage and Nardi [ABN23] but uses mutual correlated agreement (MCA) to analyze folding rounds. In contrast, our two references use a weaker notion known as *correlated agreement* (CA), or plain correlated agreement to avoid confusion. Their proofs achieve the same bounds as ours, however we consider their reasoning to be conceptually harder to grasp. The main difference lies in the guarantee that folding provides in the case when are challenges are safe (*i.e.*, state remains 0):

- MCA: if all green subsets are large enough, then agreement propagates upwards through green nodes.
- CA: if all green subsets are large enough, then agreement propagates upwards through layers at unknown nodes.

The latter loses information about where the agreement is located. Furthermore, agreement may not be aligned with green nodes. Overcoming these hurdles requires the introduction of additional tools, notably a notion of *weighted* distance (rather than standard Hamming distance) and a correlated agreement theorem for this weighted distance.

Do we gain anything? One way wonder whether using MCA rather than CA allows us to prove that FRI has more bits of security per query. Unfortunately not; as stated above, our upper bounds on the round errors is identical to previous analyses. On the other hand, we can prove that FRI satisfies stronger properties. As suggested by our case analysis, FRI exhibits two modes of operations. If there exists a layer for which the green subset W_i is small, then the verifier rejects with high probability; this is the guarantee we are already familiar with. On the other hand, if all green subsets are large, then we know that agreement propagates up through green nodes. Furthermore, an accepting transcript is one for which all verifier queries of f_0 fall on green nodes. Therefore, using our analysis, we can establish that there exists a codeword $u_0 \in \mathcal{C}_0$ such that for every query s in an accepting transcript, $f_0(s) = u_0(s)$. This property is a generalization of the opening-consistency property defined and proven in FRIDA [HSW24] to parameter settings beyond the unique-decoding radius.

Other proofs. We briefly survey other security proofs of FRI. The ethSTARK documentation [Sta21] shows that the analysis given by Ben-Sasson *et al.* [BCI+20] is in fact a proof of the round-by-round soundness of FRI. The security of FRI is also discussed in works that introduce new frameworks as a means of illustrating the power of those frameworks. Attema, Fehr and Resch [AFR23] prove that an interactive proof derived from FRI (using Merkle commitments to replace the prover's oracles) meets their notion of generalized special soundness. Evans and Angeris [EA23] prove the security of a variant of FRI using only the tools of linear algebra.

In both cases, the analyses provide looser bounds and are constrained to a smaller set of FRI parameters. We however wish to highlight that all the operations involved in the FRI protocol (encoding, folding and interpolation) are linear relations and are well-suited to be analyzed using linear algebra.

Acknowledgements

We thank Mark Simkin and Ignacio Manzur for early discussions on the topic. We also thank the 2025 cohort of the Ethereum Foundation's PQ Workshop for motivating the publication of this manuscript. Thank you to Ariel Gabizon for carefully examining our proof and catching a mistake in an earlier draft of this work.

Use of AI Tools

We recognize and disclose the use of AI tools to generate diagrams and for occasional grammar checking. The rest of the work is entirely our own.

References

[ABN23] Daniel Augot, Sarah Bordage, and Jade Nardi. Efficient multivariate low-degree tests via interactive oracle proofs of proximity for polynomial codes. *Des. Codes Cryptogr.*, 91(3):1111–1151, 2023.

[ACFY24a] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. STIR: Reed-Solomon proximity testing with fewer queries. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part X*, volume 14929 of *Lecture Notes in Computer Science*, pages 380–413. Springer, 2024.

[ACFY24b] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: Reed–Solomon proximity testing with super-fast verification. Cryptology ePrint Archive, Paper 2024/1586, 2024.

[ACY23] Gal Arnon, Alessandro Chiesa, and Eylon Yogev. IOPs with inverse polynomial soundness error. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 752–761. IEEE, 2023.

[AFR23] Thomas Attema, Serge Fehr, and Nicolas Resch. Generalized special-sound interactive proofs and their knowledge soundness. In Guy N. Rothblum and Hoeteck Wee, editors, *Theory of Cryptography - 21st International Conference, TCC 2023, Taipei, Taiwan, November 29 - December 2, 2023, Proceedings, Part III*, volume 14371 of *Lecture Notes in Computer Science*, pages 424–454. Springer, 2023.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[BCI+20] Eli Ben-Sasson, Dan Carmon, Yuval Ishai, Swastik Kopparty, and Shubhangi Saraf. Proximity gaps for reed-solomon codes. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 900–909. IEEE, 2020.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 31–60. Springer, Berlin, Heidelberg, October / November 2016.

[BGK+23] Alexander R. Block, Albert Garreta, Jonathan Katz, Justin Thaler, Pratyush Ranjan Tiwari, and Michal Zajac. Fiat-Shamir security of FRI and related SNARKs. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology - ASIACRYPT 2023 - 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, December 4-8, 2023, Proceedings, Part II*, volume 14439 of *Lecture Notes in Computer Science*, pages 3–40. Springer, 2023.

[BS05] Eli Ben-Sasson and Madhu Sudan. Simple PCPs with poly-log rate and query complexity. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 266–275. ACM Press, May 2005.

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-Shamir: from practice to theory. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1082–1090. ACM Press, June 2019.

[EA23] Alex Evans and Guillermo Angeris. Succinct proofs and linear algebra. Cryptology ePrint Archive, Paper 2023/1478, 2023.

[Hol19] Justin Holmgren. On round-by-round soundness and state restoration attacks. Cryptology ePrint Archive, Report 2019/1261, 2019.

[HSW24] Mathias Hall-Andersen, Mark Simkin, and Benedikt Wagner. FRIDA: data availability sampling from FRI. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part VI*, volume 14925 of *Lecture Notes in Computer Science*, pages 289–324. Springer, 2024.

[Sta21] StarkWare. ethSTARK documentation. Cryptology ePrint Archive, Paper 2021/582, 2021.

[Whi25a] Whiteboard Sessions and Dan Boneh. Zk whiteboard sessions – S2M7: FRI and proximity proofs (part 1). https://www.youtube.com/watch?v=MBDBrEr2XQg, 2025. YouTube video, accessed 10 October 2025.

[Whi25b] Whiteboard Sessions and Dan Boneh. Zk whiteboard sessions – S2M8: FRI and proximity proofs (part 2). https://www.youtube.com/watch?v=CWbx_rnj7LI, 2025. YouTube video, accessed 10 October 2025.

[ZCF24] Hadas Zeilberger, Binyi Chen, and Ben Fisch. Basefold: Efficient field-agnostic polynomial commitment schemes from foldable codes. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part X*, volume 14929 of *Lecture Notes in Computer Science*, pages 138–169. Springer, 2024.