

How to Fold Linear Error-Correcting Codes with Optimal Proximity Gaps

Zhe Li¹, Chaoping Xing², Yizhou Yao², Chen Yuan², Ruiqi Zhu²

¹ Institute of Mathematics and Interdisciplinary Sciences,
Xidian University, Xi'an, China
`lizh0048@e.ntu.edu.sg`

² School of Computer Science,
Shanghai Jiao Tong University, Shanghai, China
`{xingcp,yaoyizhou0620,chen_yuan,sjtuzrq7777}@sjtu.edu.cn`

Abstract. Folding is a core technique in building efficient code-based polynomial commitment schemes with polylogarithmic proof size and verification. To date, there are two families of linear error-correcting codes that have been known to allow folding, i.e., Reed-Solomon (RS) codes and foldable codes. However, neither admits optimal proximity gaps, which constitutes a fundamental bottleneck in the resulting proof sizes.

In this work, we give the first folding scheme that enjoys optimal proximity gaps. Specifically, we construct IOPPs (interactive oracle proofs of proximity) for both folded RS (FRS) codes and univariate multiplicity (UM) codes, which achieve optimal proximity gaps $1 - R - \varepsilon$ as shown in a recent work of Goyal and Guruswami (STOC 2026), where R is the code rate and $0 < \varepsilon < 1 - R$. Our key observation is that the typical even/odd folding commutes with polynomial reduction for suitably paired moduli, enabling consistency checks via the Chinese remainder theorem. This implies a new generic folding framework that recovers the FRI folding for RS codes and enables efficient folding for FRS and UM codes. We further refine the optimal proximity gap analysis for FRS and UM codes, yielding tighter concrete soundness bounds.

Our IOPPs have $O(N)$ prover time and oracle proof size, together with $O(\lambda \log N)$ query complexity and verification time, where N is the block length and λ is the security parameter. These bounds match the asymptotic complexity of FRI while attaining optimal proximity gaps and allowing smaller underlying fields at the same security level. E.g., a 128-bit field suffices for 100-bit security.

Table of Contents

1	Introduction	2
1.1	Our Contributions	3
1.2	Technical Overview	5
2	Preliminaries	11
2.1	Coding Theoretic and Proximity Gap Notions	11
2.2	Interactive Oracle Proofs of Proximity	13
3	IOPP with Optimal Proximity Gaps	14
3.1	A Generic Folding Scheme	14
3.2	IOPP for Folded Reed-Solomon Codes	17
3.3	IOPP of Univariate Multiplicity Codes	24
3.4	Concrete Proof Size	26
4	Sharp Proximity Gaps for FRS Codes and UM Codes	27
4.1	Optimal Proximity Gaps for Foldable FRS	35
A	Folding Univariate Multiplicity Codes from Derivatives	41

1 Introduction

A polynomial commitment scheme (PCS) allows a prover to commit to a polynomial $f(x)$ of degree at most k , and later open an evaluation of $f(x)$ at some point α together with a short proof guaranteeing honest opening. PCS has been a central cryptographic primitive in constructing efficient SNARKs (succinct non-interactive arguments of knowledge), which have numerous real-world applications, for example, verifiable computation, blockchain, and etc.

In the past decade, we have seen a great success of building PCSs from linear error-correcting codes [AHIV17, BCG⁺17, BBHR18, BCL22, RR22, XZS22, GLS⁺23, ZCF24, BFK⁺24, ACFY24, BCF⁺25, ACFY25], which give many promising advantages, such as practical efficiency, plausible post-quantum security, and transparent setup. In a bird’s eye view, there are three paradigms. The first [BBHR18, ACFY24, ZCF24, ACFY25], originating from the seminal protocol FRI based on Reed-Solomon codes [BBHR18], offers poly-logarithmic proof size and verification, and has been generalized to linear codes with a “folding” property. The second features code-agnosticism [AHIV17, BCL22, GLS⁺23, BFK⁺24], and thus applies to linear codes with fast encoding algorithms and implies faster proving. However, their proof size is inherently sublinear. Lastly, the code-switch technique [RR22, XZS22, BCF⁺25] allows to compose the aforementioned two approaches together, achieving the best of the two worlds. That is, the proof size is dominated by the first FRI-style scheme, while the proving time depends on the encoding efficiency in the second scheme.

A major goal in the literature is to reduce the proof size, while maintaining efficiency on other metrics. This motivates a line of works studies the proximity gap [AHIV17, BGKS20, BCI⁺20, BCH⁺26, GG26], an intriguing phenomenon of

linear error-correcting codes. Roughly speaking, the proximity gap phenomenon for a code $\mathcal{C} \subset \mathbb{F}_q^N$ states that for two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^N$, if there are sufficiently many linear combinations $\mathbf{u} + \alpha \cdot \mathbf{v}$ that are δ -close to $\mathcal{C}$ in relative Hamming distance where $\alpha \in \mathbb{F}_q$ and $\delta \in (0, 1)$ is called the proximity gap, then so are both $\mathbf{u}$ and $\mathbf{v}$. Proving a larger δ for $\mathcal{C}$ directly improves performances of PCSs that are built from $\mathcal{C}$. Notably δ has a natural upper bound $\delta < 1 - R$, where R is the rate of $\mathcal{C}$. Although significant progress on this direction has been made recently [BCH⁺26, GG26], the proximity gap of linear codes that have been used to build PCSs is still far from being optimal. For instance, Reed-Solomon codes are only known to approach the Johnson radius [BCH⁺26], i.e., $\delta < 1 - \sqrt{R}$, and there are significant barriers for proving the optimal proximity gap for RS codes due to current understanding of its list-decoding capacity. While for codes with fast encoding algorithms, one can only use $\delta = (1 - R)/3$ in unique decoding regime or the so-called 1.5 Johnson radius in list-decoding regime [BGKS20], i.e., $\delta < 1 - \sqrt[3]{R}$.

Linear error-correcting codes that attain optimal proximity gaps have not been known until a recent breakthrough of Goyal and Guruswami [GG26]. In fact, they show many subspace-design codes enjoy this property, including folded RS codes, univariate multiplicity codes, random RS codes and random linear codes. However, this does not mean we have reached the destination: it remains unclear how to obtain PCSs with better concrete efficiency from these codes. An insightful reader may feel confused here, as we have already known code-agnostic PCS templates and with the code-switch technique, one can further achieve poly-logarithmic proof size. We would like to emphasize that these codes are only feasible with Brakedown’s framework [GLS⁺23] and code-switching [RR22], but not practical. The efficiency of the resulting schemes would be crucially limited, as these codes’ encoding algorithms are much less efficient than known fast encodable codes [Spi96, BFK⁺24, BCF⁺25, GNR26, KPR⁺26]. Instead, the hope should lie at the FRI-framework, where known foldable codes all need quasi-linear time for encoding, for which folded RS codes and univariate multiplicity codes satisfy as well.

We therefore raise the following question:

Can we fold linear error-correcting codes (e.g., folded Reed-Solomon codes, and univariate multiplicity codes) with optimal proximity gaps?

1.1 Our Contributions

In this work, we answer the above question in the affirmative by constructing FRI style IOPPs (interactive oracle proofs of proximity) for folded Reed-Solomon (FRS) codes and univariate multiplicity (UM) codes with optimal proximity gaps $1 - R - \varepsilon$, where ε is an arbitrarily small constant. In fact, our approach generalizes the FRI framework to the family of polynomial ideal codes and we also give a refined proximity gap analysis, yielding sharper soundness results for our IOPPs.

A Generic Folding Framework. We develop an algebraic *folding* framework for polynomial ideal codes, in which a message polynomial f is encoded by its residues

modulo a family of pairwise coprime polynomials $e_1, \dots, e_N$ [GSS00, BHKS21]. This viewpoint captures RS, FRS, and UM codes through different choices of polynomial moduli. The key observation is that the classical even/odd folding operation commutes with reduction modulo polynomials. In more detail, let Fold denote the fold operation that given a challenge γ reduces a polynomial $f(x)$ to a polynomial $f_\gamma(y)$ in half degree, i.e., $f_\gamma(y) \leftarrow \text{Fold}(f(x), \gamma)$. For polynomials $h(x^2)$ and $h(y)$, we establish that

$$f_\gamma(y) \bmod h(y) = \text{Fold}(f(x) \bmod h(x^2), \gamma)(y) \quad (1)$$

Assume $h(x^2)$ is the product of two coprime polynomial moduli e_i, e_j , i.e., $h(x^2) = e_i(x) \cdot e_j(x)$. Then Chinese Remainder Theorem (CRT) allows to reconstruct $g(x) := f(x) \bmod h(x^2)$ from

$$(f(x) \bmod e_i(x)) \quad \text{and} \quad (f(x) \bmod e_j(x)).$$

Now suppose the verifier is given two oracles $f(x), f_\gamma(y)$ and he can perform a “modulo” query to these oracles. That is, by querying $f(x)$ at some $e_i(x)$, he receives $f(x) \bmod e_i(x)$. Having reconstructed $g(x)$, the verifier can locally compute $\text{Fold}(g(x), \gamma)$. Together with query of $f_\gamma(y)$ at $h(y)$, he can check the consistency of $f(x), f_\gamma(y)$ according to Eq.(1). As $f_\gamma(y)$ has half degree of $f(x)$, the oracle size is halved, which implies a generic folding scheme.

Our framework recovers standard FRI folding for RS codes and extends to FRS and UM codes. For RS codes, $e_1, \dots, e_N$ are linear polynomials, while for FRS codes and UM codes, they are in the form of $\prod_{j=0}^{m-1} (x - \zeta^j a)$ and $(x - a)^m$, respectively. With suitable evaluation set, folding can be applied recursively while preserving the code rate. Combining with the refined proximity gap results below yields the desired IOPPs. Note that the number of query is $O(\lambda \log N)$ as in FRI, where λ denotes the security parameter. While each query response consists of $m = \deg(e_i)$ elements rather than 1 element as in FRI.

Proximity Gaps. Building on [GG26], we refine the proximity gap analysis for FRS and UM codes. The asymptotically optimal radius $1 - R - \varepsilon$ and the asymptotic soundness order remain unchanged, while our analysis substantially reduces the required folding parameter and gives more precise concrete soundness bounds.

Theorem 1.1 (Informal; see Theorem 4.14). *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be either an m FRS or an order m UM code of rate R . Fix $0 < \varepsilon < 1 - R$. Under the conditions of Theorem 4.14, if $m \geq \frac{4R(2-R)}{\varepsilon^2} + \frac{4(1-R)}{\varepsilon}$, then $\mathcal{C}$ has mutual correlated agreement at radius $1 - R - \varepsilon$ with soundness $O\left(\frac{N/\varepsilon + 1/\varepsilon^3}{q}\right)$.*

Remark 1.2. For the same target radius $1 - R - \varepsilon$, the result of [GG26] achieves the same soundness order $O\left(\frac{N/\varepsilon + 1/\varepsilon^3}{q}\right)$. However, it requires the folding parameter $m > 4 \left\lceil \frac{2}{\varepsilon} \right\rceil^2$. In comparison, our analysis requires a substantially smaller folding parameter while preserving the same radius and soundness order.

Our IOPPs for FRS and UM codes have proof sizes comparable to those of FRI. Under the soundness bounds above, our concrete instantiations achieve 100-bit security over a 128-bit field, compared with a 192-bit field for the corresponding FRI instantiations. Refer to [Table 3](#) and [Table 4](#) for details. Compared with [\[GG26\]](#), our sharper bounds allow a smaller m with identical soundness. The improvement is particularly visible for certain large circuit instantiations, where our bounds also permit a smaller field at the same proximity radius and security level.

Towards IOPPs for FRS/UM codes with smaller proof size, the main bottleneck lies at $m = O(1/\varepsilon^2)$ for achieving optimal proximity gaps, as each query will return a symbol of length m . Very recently, Jeronimo shows an exact support MCA theorem for ordinary RS codes [\[Jer26, Theorem 1.2\]](#), which can be used to reduce m to $O(1/\varepsilon)$. However, this improvement comes at the cost of a prohibitively large soundness error bound. Thus, obtaining optimal proximity gaps with small m and sufficiently small soundness error for practical parameter choices remains an open direction and we leave it as future work.

1.2 Technical Overview.

FRI overview. IOPP (Interactive Oracle Proof of Proximity) plays a crucial role in building polynomial commitments. An IOPP allows a prover $\mathcal{P}$ to convince a verifier $\mathcal{V}$ that a vector w is close to a given code $\mathcal{C}$, e.g., $\Delta(w, \mathcal{C}) \leq \delta$, where $\mathcal{V}$ has oracle access to w , and $\delta \in \{0, 1\}$. IOPP with logarithmic query complexity has been known to exist for “foldable” linear codes. At a high-level, folding allows to reduce the statement $\Delta(w, \mathcal{C}) \leq \delta$ to a new statement $\Delta(w', \mathcal{C}') \leq \delta'$ of smaller size through querying a few symbols of w, w' . By repeating the folding procedure for a sufficient number of times, the statement size is reduced to constant and thus can be verified directly.

The celebrated work FRI [\[BBHR18\]](#) gives an efficient IOPP for Reed-Solomon codes, which is the starting point of our new IOPP and we briefly overview below. Suppose $\mathcal{L}$ is a multiplicative subgroup of $\mathbb{F}_q^*$ of size $N = 2^n$. An RS code over $\mathcal{L}$ with rate R , denoted by $\text{RS}[\mathcal{L}, R]$, is the collection of polynomial evaluations $(f(\alpha_1), \dots, f(\alpha_N)) \in \mathbb{F}_q^N$, where $f(x)$ is a polynomial over $\mathbb{F}_q$ with degree less than RN and $\alpha_i \in \mathcal{L}$. For convenience, we also view $f \in \text{RS}[\mathcal{L}, R]$ as a function $\mathcal{L} \rightarrow \mathbb{F}_q$. There exists a folding map

$$\begin{aligned} \text{Fold} : \text{RS}[\mathcal{L}, R] \times \mathbb{F}_q &\rightarrow \text{RS}[\mathcal{L}^2, R], \\ (f, \gamma) &\mapsto f_\gamma := \text{Fold}(f, \gamma), \end{aligned}$$

where $\mathcal{L}^2, \mathcal{L} \subset \mathbb{F}$ with $|\mathcal{L}^2| = |\mathcal{L}|/2$, such that:

1. If $f \in \text{RS}[\mathcal{L}, R]$, then for every $\gamma \in \mathbb{F}_q$, $f_\gamma \in \text{RS}[\mathcal{L}^2, R]$;
2. If f is δ -far from $\text{RS}[\mathcal{L}, R]$ for some $\delta \in (0, 1 - \sqrt{R})$, then f_γ is δ -far from $\text{RS}[\mathcal{L}^2, R]$, with a high probability over the choice of γ .
3. There exists an efficient algorithm that given γ , and oracle access to f , can output $f_\gamma(\alpha)$ for any $\alpha \in \mathcal{L}^2$, by making some queries to f .

Intuitively, **Property 1** guarantees completeness and **Property 3** allows the verifier to check the consistency of the two oracles f, f_γ , and thus the prover is forced to send the correct message $f_\gamma := \text{Fold}(f, \gamma)$. By **Property 2**, one can conclude that f is δ -close to $\text{RS}[\mathcal{L}, R]$ as long as f_γ is δ -close to $\text{RS}[\mathcal{L}^2, R]$. Thus the proximity test problem for $f \in \text{RS}[\mathcal{L}, R]$ is reduced to $f_\gamma \in \text{RS}[\mathcal{L}^2, R]$.

Let us now look into the folding map. It is well-known that every polynomial $f(x)$ of degree $< RN$ can be uniquely written as

$$f(x) = f_0(x^2) + x \cdot f_1(x^2),$$

where f_0, f_1 are two polynomials over $\mathbb{F}_q$ with degree less than $RN/2$. The folding map is defined in the following way:

$$\text{Fold}(f, \gamma)(y) := f_0(y) + \gamma \cdot f_1(y).$$

Here, to make it more clear, we use variable y instead of reusing x to define the folded polynomial. In particular, $\mathcal{L}^2 = \{\alpha^2 \mid \alpha \in \mathcal{L}\}$. Since $\mathcal{L}$ is a multiplicative subgroup of order $N = 2^n$ in $\mathbb{F}_q^*$, $\mathcal{L}^2$ is a multiplicative subgroup as well and $|\mathcal{L}^2| = N/2$. It is easy to see the above **Fold** definition satisfies **Property 1**.

For the latter two properties, we remark that oracle access to $f \in \text{RS}[\mathcal{L}, R]$ essentially implies access to two virtual oracles $f_0, f_1 \in \text{RS}[\mathcal{L}^2, R]$. This is because querying f_0, f_1 at some α^2 can be emulated by querying f at $\alpha, -\alpha$, and compute $f_0(\alpha^2)$ and $f_1(\alpha^2)$ as

$$f_0(\alpha^2) = \frac{f(\alpha) + f(-\alpha)}{2} \text{ and } f_1(\alpha^2) = \frac{f(\alpha) - f(-\alpha)}{2\alpha}.$$

Thus, through queries to $f \in \text{RS}[\mathcal{L}, R]$, the verifier can compute

$$f_\gamma(\alpha^2) := f_0(\alpha^2) + \gamma \cdot f_1(\alpha^2)$$

for any $\alpha^2 \in \mathcal{L}^2$. Moreover, if f is δ -far from $\text{RS}[\mathcal{L}, R]$, then at least one of f_0, f_1 would be δ -far from $\text{RS}[\mathcal{L}^2, R]$. Combining with the state-of-the-art proximity gap results of RS codes, one immediately gets **Property 2**.

A generic folding framework. Now we are ready to show how to extend the above folding framework from Reed-Solomon codes to FRS codes and UM codes. To do so, we first introduce a generic framework from the CRT viewpoint.

Recall that in FRI, the verifier needs to check consistency of the two oracles f, f_γ . This is done by querying f_γ at a random evaluation point $\alpha^2 \in \mathcal{L}^2$, and querying f at $\alpha, -\alpha$. By Lagrange interpolation on $(\alpha, f(\alpha)), (-\alpha, f(-\alpha))$ the verifier gets a linear polynomial

$$g(x) := \frac{f(\alpha) + f(-\alpha)}{2} + \frac{f(\alpha) - f(-\alpha)}{2\alpha}x = f_0(\alpha^2) + x \cdot f_1(\alpha^2), \quad (2)$$

which satisfies $g(\gamma) = f_\gamma(\alpha^2)$. Now, we revisit the above procedure. First of all, we note that evaluating f at some point α can be represented as

$$f(\alpha) = f(x) \mod (x - \alpha).$$

Similarly, $f(-\alpha) = f(x) \bmod (x + \alpha)$. Jumping ahead, this viewpoint is crucial for extending FRI to IOPP of FRS/UM codes, where each symbol is no longer one polynomial evaluation, but a vector of evaluations. By the definition of $g(x)$ as in [Eq. \(2\)](#), we know that $g(\alpha) = f(\alpha)$, and $g(-\alpha) = f(-\alpha)$. Thus,

$$g(x) \bmod (x - \alpha) = f(x) \bmod (x - \alpha),$$

and similarly

$$g(x) \bmod (x + \alpha) = f(x) \bmod (x + \alpha).$$

By the well-known Chinese Remainder Theorem (CRT), there exists a unique polynomial $r(x)$ of degree less than 2 such that

$$r(x) \equiv f(x) \bmod (x - \alpha)(x + \alpha).$$

In fact, $r(x)$ is exactly the polynomial $g(x)$! Namely,

$$g(x) = f(x) \bmod (x^2 - \alpha^2).$$

Therefore, one can view querying f at $\{\pm\alpha\}$ and then computing $g(x)$ as just one operator: $\bmod (x^2 - \alpha^2)$.

Moreover, as $g(x)$ is a linear polynomial, we observe that evaluating g at γ is equivalent to apply $\text{Fold}(\cdot, \gamma)$, i.e., $g(\gamma) = \text{Fold}(g(x), \gamma)$. Eventually, we obtain a commutative diagram for $\bmod$ and Fold :

$$\begin{array}{ccc} f(x) & \xrightarrow{\bmod (x^2 - \alpha^2)} & g(x) \\ \downarrow \text{Fold} & & \downarrow \text{Fold} \\ f_\gamma(y) & \xrightarrow{\bmod (y - \alpha^2)} & f_\gamma(\alpha^2) = g(\gamma) \end{array}$$

By the above diagram, it is clear that checking consistency of f, f_γ reduces to checking $f_\gamma(\alpha^2) = g(\gamma)$. In the following, we show that the above commutative diagram also extend to FRS codes and UM codes, the difference would lie at the modulus polynomial.

Folding FRS codes. Let $\mathcal{L} := \{\alpha_1, \dots, \alpha_N\}$ be a multiplicative subgroup of $\mathbb{F}_q^*$ of order $N = 2^n$, and $\zeta \in \mathbb{F}_q$ such that $\zeta^j \alpha_i$ are distinct for $j \in [0, m-1], i \in [1, N]$. Assume for now ζ is primitive. Define the cosets $\zeta^j \mathcal{L} := \{\zeta^j \alpha \mid \alpha \in \mathcal{L}\}$ for each $j \in [0, m-1]$. An FRS code, denoted by $\text{FRS}[\mathcal{L}, \zeta, R]$, is the collection of

$$(\vec{f}_1, \dots, \vec{f}_N) \in (\mathbb{F}_q^m)^N, \text{ where } \vec{f}_i = (f(\zeta^0 \alpha_i), \dots, f(\zeta^{m-1} \alpha_i)) \in \mathbb{F}_q^m, \quad (3)$$

and f is a polynomial over $\mathbb{F}_q$ of degree less than RmN . Note that if $m = 1$, FRS codes are exactly RS codes. Intuitively, an FRS codeword may be viewed as an RS codeword in which each symbol is replaced by a column vector of m evaluations. This in fact makes folding FRS codes challenging: for consistency check it seems that one should be able to interpolate a polynomial from 2 symbols of the codeword, which corresponds to $2m$ evaluations in FRS setting (rather

than 2 in RS setting), and it is hard to see how it relates to the folded polynomial f_γ . Our new framework gives a clear characterization to this issue.

Since we are still dealing with polynomial evaluations, we fold polynomials in the same way as in FRI. The remaining thing is the consistency check of f, f_γ , where $f \in \text{FRS}[\mathcal{L}, \zeta, R]$, $f_\gamma = \text{Fold}(f, \gamma) \in \text{FRS}[\mathcal{L}, \zeta^2, R]$ with $\mathcal{L}^2 := \{\alpha_i^2 \mid i \in [1, N]\}$. Here our core observation is that for each $i \in [1, N]$ there exists an isomorphism between the symbol $\vec{f}_i$ and the residue polynomial

$$f^{(i)}(x) := f(x) \mod \left(\prod_{j=0}^{m-1} (x - \zeta^j \alpha_i) \right).$$

The isomorphism is essentially induced by CRT. Because every $\zeta^j \alpha_i$ is distinct and $f^{(i)}(x) = f(x)$ for each $x \in \{\zeta^j \alpha_i\}_{j \in [0, m-1]}$.

Suppose $i' \in [1, N]$ such that $\alpha_{i'} = -\alpha_i$, which is always the case since $\{\alpha_1, \dots, \alpha_N\}$ is a multiplicative subgroup of order $N = 2^n$. Now it is clear to see that the polynomial of degree $< 2m$ that agrees with $f^{(i)}(x)$ on $\{\zeta^j \alpha_i\}_{j \in [0, m-1]}$ and $f^{(i')}(x)$ on $\{-\zeta^j \alpha_i\}_{j \in [0, m-1]}$, is exactly the residue polynomial

$$g(x) = f(x) \mod \prod_{j=0}^{m-1} (x^2 - \zeta^{2j} \alpha_i^2).$$

Correctness directly follows that $g(x) \equiv f(x) \mod \left(\prod_{j=0}^{m-1} (x - \zeta^j \alpha_i) \right)$, and $g(x) \equiv f(x) \mod \left(\prod_{j=0}^{m-1} (x + \zeta^j \alpha_i) \right)$, and CRT. Although unlike FRI here $g(x)$ is no longer a linear polynomial, we can still apply Fold. Since mod commutes with Fold, we prove that the folded polynomial, denoted by $g_\gamma(y)$, equals to

$$f_\gamma(y) \mod \prod_{j=0}^{m-1} (y - \zeta^{2j} \alpha_i^2),$$

i.e., the symbol of f_γ 's indexed by α_i^2 . This gives the following commutative diagram:

$$\begin{array}{ccc} f(x) & \xrightarrow{\mod \prod_{j=0}^{m-1} (x^2 - \zeta^{2j} \alpha_i^2)} & g(x) \\ \downarrow \text{Fold} & & \downarrow \text{Fold} \\ f_\gamma(y) & \xrightarrow{\mod \prod_{j=0}^{m-1} (y - \zeta^{2j} \alpha_i^2)} & g_\gamma(y) \end{array}$$

In sketch, for the consistency check it suffices for the verifier to randomly pick a pair $(\alpha_i, -\alpha_i)$ and query $f(x)$ at the column corresponding to $\pm \alpha_i$. This allows the verifier to compute $g(x)$ via CRT and further obtain $g_\gamma(y) = \text{Fold}(g(x), \gamma)$, which equals to the polynomial obtained by querying $f_\gamma(y)$ at the column corresponding to α_i^2 .

The above verification procedure would take $O(m \log^2 m)$ time due to CRT reconstruction, assuming $\mathbb{F}_q$ is a FFT-friendly field. In the following, we show

that $g(x)$ has not to be reconstructed explicitly, leading to $O(m)$ verification time. To achieve this, we utilize the fact that the modulus polynomials

$$\prod_{j=0}^{m-1} (x^2 - \zeta^{2j} \alpha_i^2) \quad \text{and} \quad \prod_{j=0}^{m-1} (y - \zeta^{2j} \alpha_i^2)$$

split completely into linear factors. This essentially allows to check consistency of f, f_γ row-by-row, which in intuition, is equivalent to fold m RS codewords simultaneously. Indeed, for every $j \in [0, m-1]$, we have

$$f_\gamma(\zeta^{2j} \alpha_i^2) = \frac{f(\zeta^j \alpha_i) + f(-\zeta^j \alpha_i)}{2} + \frac{\gamma}{2\zeta^j \alpha_i} (f(\zeta^j \alpha_i) - f(-\zeta^j \alpha_i)).$$

Hence, each entry of the folded codeword is determined directly by the corresponding two symbols of the original codeword. The consistency check can therefore be performed row-wise in $O(m)$ field operations, without polynomial interpolation and CRT reconstruction.

One caveat is that the optimal proximity gaps for FRS codes was proved in terms of a *primitive* ζ . However, folding changes both the evaluation set and the multiplier, replacing ζ by ζ^{2^r} after r rounds. Note that ζ^{2^r} is no longer primitive as we need $\mathbb{F}_q^*$ to have 2-smooth multiplicative subgroups for FFT. Hence, we need to ensure that the codes produced by repeated folding still admit optimal proximity gaps. To do so, we choose a ζ with sufficiently large odd multiplicative order. Since ζ has odd order, ζ^{2^r} has the same order as ζ whereas α^{2^r} has order $N/2^r$. Their generated subgroups therefore intersect only at 1, which guarantees that the evaluation points $\alpha^{2^r i} \zeta^{2^r j}$ are all distinct. Thus, in every round we still get a properly defined m folded Reed-Solomon code, as shown in [Lemma 4.16](#). To prove that our FRS codes indeed achieve the optimal proximity gap, it remain to show that it is a subspace design code. [Lemma 4.17](#) prove this fact by showing that $f_1, \dots, f_d$ are $\mathbb{F}_q$ -linearly independent if and only if they form a folded Wronskian matrix with nonzero determinant. This suffices to prove the subspace design property.

During the folding phase, the message length and the block length decrease by the same factor, while the FRS parameter m remains unchanged. Hence the rate remains R throughout folding. Combining the subspace design bound with list decodability allows us to apply [Theorem 4.14](#) at every round. Under its parameter conditions, we therefore obtain mutual correlated agreement, and hence proximity gaps at radius $1 - R - \varepsilon$, throughout the folding phase; see [Corollary 4.18](#).

Folding UM codes. Let $\mathcal{L} = \{\alpha_1, \dots, \alpha_N\}$ be a multiplicative subgroup of $\mathbb{F}_q^*$ with size $N = 2^n$. A univariate multiplicity (UM) code, denoted by $\text{UM}[\mathcal{L}, m, R]$, is the collection of

$$(\vec{f}_1, \dots, \vec{f}_N) \in (\mathbb{F}_q^m)^N, \text{ where } \vec{f}_i = (f(\alpha_i), f^{(1)}(\alpha_i), \dots, f^{(m-1)}(\alpha_i)),$$

and f is a polynomial over $\mathbb{F}_q$ of degree less than RmN . Here $f^{(j)}(x)$ denotes the j -th derivative of $f(x)$.

Similarly, if $m = 1$, UM codes are exactly RS codes and we hope to borrow the fold map again. However note that each symbol of a UM codeword consists of evaluations of m polynomials $f(x), f^{(1)}(x), \dots, f^{(m-1)}(x)$, and it is not hard to see that **Fold** does not commute with derivatives. That is,

$$\text{Fold}(f(x))^{(j)} \neq \text{Fold}(f^{(j)}(x)).$$

This means that unlike FRS codes, here we can not check the equality row-by-row. Still, we note that in the CRT viewpoint, for every $i \in [1, N]$ there exists an isomorphism between $\vec{f}_i$ and the residue polynomial [BHK21, Section 3.1]

$$f(x) \bmod (x - \alpha_i)^m.$$

Thus, the following commutative diagram holds:

$$\begin{array}{ccc} f(x) & \xrightarrow{\bmod (x^2 - \alpha_i^2)^m} & g(x) \\ \downarrow \text{Fold} & & \downarrow \text{Fold} \\ f_\gamma(y) & \xrightarrow{\bmod (y - \alpha_i^2)^m} & g_\gamma(y) \end{array}$$

Very similarly, the consistency check of f, f_γ proceeds as follows: the verifier randomly pick α_i^2 and query $f(x)$ at the column corresponding to $\alpha_i, -\alpha_i$, i.e.,

$$f(x) \bmod (x - \alpha_i)^m \quad \text{and} \quad f(x) \bmod (x + \alpha_i)^m.$$

This allows the verifier to compute $g(x)$ and further obtain $g_\gamma(y) = \text{Fold}(g(x), \gamma)$, which should equal to the polynomial obtained by querying $f_\gamma(x)$ at the column corresponding to α_i^2 , i.e., $f_\gamma(y) \bmod (y - \alpha_i^2)^m$. We show that reconstructing $g(x)$ can be done in $O(m \log m)$ operations in $\mathbb{F}_q$. More details are referred to [Section 3.3](#). In particular, we also give a direct consistency check from the derivative perspective in [Appendix A](#).

Proximity Gaps for FRS and UM Codes. Our proof follows the curve decoding approach of [GG26]. Consider a received word curve $\mathbf{u}_\alpha = \sum_{j=0}^\ell \mathbf{u}_j \alpha^j$ and for every parameter α for which $\mathbf{u}_\alpha$ is ρ close to $\mathcal{C}$, choose a nearby codeword $f(\alpha) \in \mathcal{C}$. The goal is to show that many of these nearby codewords lie on a single degree ℓ codeword curve. For each coordinate i , the possible values of $(\mathbf{u}_\alpha)_i$ lie in the affine space $(\mathbf{u}_0)_i + \text{span}\{(\mathbf{u}_1)_i, \dots, (\mathbf{u}_\ell)_i\}$. We fix one nearby codeword $\mathbf{c}^{(0)}$ and let H be the span of the differences $\mathbf{c} - \mathbf{c}^{(0)}$ over all relevant nearby codewords $\mathbf{c}$. Using the subspace design property, we show that H has small dimension. Unlike [GG26], we work directly with the affine spaces above, rather than first enlarging them to linear spaces, which avoids an extra dimension loss.

We then prune the low dimensional space H to find a small set of coordinates S such that $H_S = \{\mathbf{0}\}$. We use the same pruning procedure as in [GG26], but keep the exact success probability $\varepsilon/(\rho \dim_{\mathbb{F}_q} H + \varepsilon)$. Since the pruning distribution depends only on H , the same set S can be chosen for many good parameters α . For these parameters, both $f(\alpha)$ and the received word $\mathbf{u}_\alpha$ agree on S . Interpolating a few of the codewords $f(\alpha)$ therefore gives a degree ℓ codeword curve whose

restriction to S agrees with every $f(\alpha)$ in this subset. Because the difference lies in H and vanishes on S , the condition $H_S = \{0\}$ forces equality on all coordinates. This yields the weak curve decodability guarantee.

The remaining steps follow [GG26]. List decodability amplifies the number of nearby parameters lying on one codeword curve, and this gives correlated agreement. For mutual correlated agreement, outside the common agreement set, the difference of the two degree ℓ curves is a nonzero polynomial in α of degree at most ℓ and hence vanish for at most ℓ parameter values at each coordinate. Keeping this exact count yields the threshold $\lfloor \ell \rho N + \ell \rfloor + 1$ and a sharper finite parameter bound. Finally, substituting the known subspace design and list decoding parameters of FRS and UM codes gives Theorem 4.14 and in particular the $1 - R - \varepsilon$ line proximity gap.

2 Preliminaries

Throughout the paper, q denotes a prime power and $\mathbb{F}_q$ denotes the finite field of q elements. When $q = p$ is prime, we also write $\mathbb{F}_p$. For a positive integer N , let $[N] := \{1, 2, \dots, N\}$.

Let Σ be an alphabet, and let $\mathbf{x} = (x_1, \dots, x_N), \mathbf{y} = (y_1, \dots, y_N) \in \Sigma^N$. Their Hamming distance and relative Hamming distance are defined by $d(\mathbf{x}, \mathbf{y}) := |\{i \in [N] : x_i \neq y_i\}|$ and $\Delta(\mathbf{x}, \mathbf{y}) := \frac{d(\mathbf{x}, \mathbf{y})}{N}$. For a nonempty set $\mathcal{Y} \subseteq \Sigma^N$, define $\Delta(\mathbf{x}, \mathcal{Y}) := \min_{\mathbf{y} \in \mathcal{Y}} \Delta(\mathbf{x}, \mathbf{y})$. In particular, if $\mathcal{C} \subseteq \Sigma^N$ is a code, then $\Delta(\mathbf{x}, \mathcal{C})$ denotes the relative distance from $\mathbf{x}$ to $\mathcal{C}$. When Σ is a vector space over $\mathbb{F}_q$, define $\text{supp}(\mathbf{x}) := \{i \in [N] : x_i \neq 0\}$ and $\text{wt}(\mathbf{x}) := |\text{supp}(\mathbf{x})|$. In this case, $d(\mathbf{x}, \mathbf{y}) = \text{wt}(\mathbf{x} - \mathbf{y})$.

For a subset $S \subseteq [N]$, we denote by $\mathbf{x}|_S := (x_i)_{i \in S}$ the restriction of $\mathbf{x}$ to the coordinates in S . If $H \subseteq \Sigma^N$ is an $\mathbb{F}_q$ linear subspace, define $H_S := \{\mathbf{h} \in H : \mathbf{h}|_S = \mathbf{0}\}$ and $H_i := H_{\{i\}}$ for $i \in [N]$. For $\mathbf{y} \in \Sigma^N$ and $\rho \in [0, 1]$, define $\mathcal{L}_{\mathcal{C}}(\mathbf{y}, \rho) := \{\mathbf{c} \in \mathcal{C} : \Delta(\mathbf{c}, \mathbf{y}) \leq \rho\}$. We say that $\mathcal{C}$ is (ρ, L) list decodable if $\max_{\mathbf{y} \in \Sigma^N} |\mathcal{L}_{\mathcal{C}}(\mathbf{y}, \rho)| \leq L$.

Finally, when Σ is an $\mathbb{F}_q$ vector space, for coefficient words $\mathbf{u}_0, \dots, \mathbf{u}_{\ell} \in \Sigma^N$ and $\alpha \in \mathbb{F}_q$, write $\mathbf{u}_{\alpha} := \sum_{j=0}^{\ell} \mathbf{u}_j \alpha^j$. We refer to the map $\alpha \mapsto \mathbf{u}_{\alpha}$ as a degree- ℓ word curve. When $\ell = 1$, we call it a word line.

2.1 Coding Theoretic and Proximity Gap Notions

We first recall the two folded polynomial evaluation code families to which our sharp proximity gap analysis applies: FRS and UM codes.

Definition 2.1 (Folded Reed-Solomon Codes [GR08]). *Let N, k, m be positive integers, and let $\mathbb{F}_q$ be a finite field with $q > mN$ and $w \in \mathbb{F}_q^*$ be an element of multiplicative order at least k . Additionally, let $\alpha_1, \alpha_2, \dots, \alpha_N \in \mathbb{F}_q$ be distinct elements such that $\alpha_i \zeta^t \neq \alpha_j$ for all $i \neq j$ and $t < m$.*

The m FRS code $\mathcal{C}_{\text{FRS}} \subseteq (\mathbb{F}_q^m)^N$ with message length k on $\mathcal{L} = \{\alpha_1, \dots, \alpha_n\}$ is given by

$$\text{FRS}_{q,N,k,m}(\mathcal{L}, \zeta) := \left\{ \left(\left[\begin{array}{c} f(\alpha_i) \\ f(\alpha_i \zeta) \\ \vdots \\ f(\alpha_i \zeta^{m-1}) \end{array} \right] \right)_{i \in [N]} \mid f \in \mathbb{F}_q[x], \deg f < k \right\}.$$

For simplicity, we also use $\text{FRS}[\mathcal{L}, \zeta, R]$ to denote an FRS code where $R = k/(mN)$ is the rate.

Definition 2.2 (Univariate Multiplicity Codes [RT97, Kop15]). Let N, k, m be positive integers and let $\mathbb{F}_q$ be a finite field with $q \geq N$. Let $\alpha_1, \alpha_2, \dots, \alpha_N \in \mathbb{F}_q$ be distinct elements. For a polynomial $f(X) \in \mathbb{F}_q[X]$, its j -th Hasse derivative $f^{(j)}(X)$ is defined by the identity $f(X + Z) = \sum_{j \geq 0} f^{(j)}(X) Z^j$.

The order m UM code $\mathcal{C}_{\text{Mult}} \subseteq (\mathbb{F}_q^m)^N$ with message length k on $\mathcal{L} = (\alpha_1, \dots, \alpha_N)$ is given by

$$\text{MULT}_{q,N,k,m}(\alpha) := \left\{ \left(\left[\begin{array}{c} f(\alpha_i) \\ f^{(1)}(\alpha_i) \\ \vdots \\ f^{(m-1)}(\alpha_i) \end{array} \right] \right)_{i \in [N]} \mid f \in \mathbb{F}_q[x], \deg f < k \right\}.$$

For simplicity, we also use $\text{FRS}[\mathcal{L}, m, R]$ to denote a UM code where $R = k/(mN)$ is the rate.

We show the structural property of the code that will be used in the pruning argument. We next recall the subspace design property, which provides the structural input for our pruning argument.

Definition 2.3 (τ subspace design code [GK16, CZ25]). Let $\tau : \mathbb{N}_{\geq 1} \rightarrow \mathbb{R}_{\geq 0}$ be a function, and let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be an $\mathbb{F}_q$ linear code. For an $\mathbb{F}_q$ linear subspace $A \subseteq \mathcal{C}$ and $i \in [N]$, define $A_i := \{a \in A : a_i = 0\}$.

We say that $\mathcal{C}$ is a τ subspace design code if, for every integer $s \geq 1$ and every $\mathbb{F}_q$ linear subspace $A \subseteq \mathcal{C}$ with $\dim_{\mathbb{F}_q} A \leq s$, we have $\sum_{i=1}^N \dim_{\mathbb{F}_q} (A_i) \leq N \cdot \tau(s) \cdot \dim_{\mathbb{F}_q} A$.

We now formalize the proximity and agreement properties that constitute our final goal. Correlated agreement and mutual correlated agreement are stronger structural notions from which proximity gaps follow.

Definition 2.4 (Proximity gaps [GG26]). An $\mathbb{F}$ linear code $\mathcal{C} \subseteq \Sigma^N$ has an $(\ell, \delta, \varepsilon, \gamma)$ proximity gap if, for every $\mathbf{u}_0, \dots, \mathbf{u}_\ell \in \Sigma^N$ and every $\delta' \leq \delta$, whenever $\Pr_{\alpha \in \mathbb{F}}[\Delta(\mathbf{u}_\alpha, \mathcal{C}) \leq \delta'] > \varepsilon$, we have $\Delta(\mathbf{u}_\alpha, \mathcal{C}) \leq \delta'/(1 - \gamma)$ for every $\alpha \in \mathbb{F}$. For $\ell = 1$, this is called a $(\delta, \varepsilon, \gamma)$ line proximity gap.

Definition 2.5 (Correlated agreement [GG26]). An $\mathbb{F}$ linear code $\mathcal{C} \subseteq \Sigma^N$ is said to have $(\ell, \delta, \varepsilon, \gamma)$ correlated agreement if, for every $\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_\ell \in \Sigma^N$ and every $\delta' \leq \delta$, $\Pr_{\alpha \in \mathbb{F}} \left[\Delta \left(\sum_{j=0}^{\ell} \alpha^j \mathbf{u}_j, \mathcal{C} \right) \leq \delta' \right] > \varepsilon$ implies that there exist $\mathbf{c}_0, \mathbf{c}_1, \dots, \mathbf{c}_\ell \in \mathcal{C}$ such that $|\{i \in [N] : \exists j \in \{0, 1, \dots, \ell\} \text{ such that } c_{j,i} \neq u_{j,i}\}| \leq \frac{\delta' N}{1-\gamma}$. When $\ell = 1$, we also say that $\mathcal{C}$ has $(\delta, \varepsilon, \gamma)$ line correlated agreement. When $\gamma = 0$, we call this perfect correlated agreement.

Definition 2.6 (Mutual correlated agreement [GG26]). An $\mathbb{F}$ linear code $\mathcal{C} \subseteq \Sigma^N$ is said to have $(\ell, \delta, \varepsilon)$ mutual correlated agreement if, for every $\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_\ell \in \Sigma^N$ and every $\delta' \leq \delta$,

$$\Pr_{\alpha \in \mathbb{F}} \left[\begin{array}{l} \exists S \subseteq [N] \text{ with } |S| \geq (1 - \delta')N, \\ \exists \mathbf{c} \in \mathcal{C} \text{ such that } \mathbf{c}|_S = \left(\sum_{j=0}^{\ell} \alpha^j \mathbf{u}_j \right)|_S, \\ \exists j \in \{0, 1, \dots, \ell\} \text{ such that } \forall \mathbf{c}' \in \mathcal{C}, \mathbf{c}'|_S \neq \mathbf{u}_j|_S \end{array} \right] \leq \varepsilon.$$

2.2 Interactive Oracle Proofs of Proximity

Let $\mathcal{R} = \{(\mathbf{x}, \mathbf{y}, \mathbf{w})\}$ be a ternary relation. An *interactive oracle proof* (IOP) for $\mathcal{R}$ allows a prover to convince a verifier that there exists a witness $\mathbf{w}$ such that $(\mathbf{x}, \mathbf{y}, \mathbf{w}) \in \mathcal{R}$. Specifically, a k -round public-coin IOP for $\mathcal{R}$ works as follows. The prover receives $(\mathbf{x}, \mathbf{y}, \mathbf{w})$ as input, while the verifier receives $\mathbf{x}$ as input and has oracle access to $\mathbf{y}$. In each round $i \in [1, k]$, the verifier will send a uniformly random challenge γ_i to the prover, who then on input $(\mathbf{x}, \mathbf{y}, \mathbf{w})$ and all previous challenges $(\gamma_1, \dots, \gamma_i)$, outputs a proof π_i to the verifier. After k rounds of interaction, the verifier makes some queries to $\mathbf{y}$ and $\pi_1, \dots, \pi_k$, and then outputs a decision bit. We denote by $\mathbf{P}$ the prover's algorithm for computing π_i 's and by $\mathbf{V}$ the verifier's decision algorithm.

In particular, we consider IOPs of proximity, in which $\mathbf{x}$ typically specifies the parameters of a code, $\mathbf{y}$ specifies an alleged codeword, and $\mathbf{w}$ is just an empty string. Informally, in IOPP the prover convinces the verifier that $\mathbf{y}$ is close to some code. The formal definition is given as follows.

Definition 2.7 (Interactive oracle proof of proximity). We say $\text{IOPP} = (\mathbf{P}, \mathbf{V})$ is a k -round public-coin IOP of proximity for a relation $\mathcal{R}$ with perfect completeness and soundness ϵ if the following holds:

– **Perfect completeness.** For each $(\mathbf{x}, \mathbf{y}, \mathbf{w})$,

$$\Pr_{\gamma_1, \dots, \gamma_k} [\mathbf{V}^{\mathbf{y}, \pi_1, \dots, \pi_k}(\mathbf{x}, \gamma_1, \dots, \gamma_k) = 1 \mid i \in [1, k], \pi_i \leftarrow \mathbf{P}(\mathbf{x}, \mathbf{y}, \mathbf{w}, \gamma_1, \dots, \gamma_i)] = 1.$$

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

$$\Pr_{\gamma_1, \dots, \gamma_k} [\mathbf{V}^{\mathbf{y}, \pi_1, \dots, \pi_k}(\mathbf{x}, \gamma_1, \dots, \gamma_k) = 1 \mid i \in [1, k], \pi_i \leftarrow \tilde{\mathbf{P}}(\gamma_1, \dots, \gamma_i)] \leq \epsilon(\mathbf{x}, \mathbf{y}).$$

When the soundness error depends only on the lengths of the inputs and on the distance δ of $\mathbf{y}$ from the language $L_{\mathbf{x}} := \{\mathbf{y}' : \exists \mathbf{w}, (\mathbf{x}, \mathbf{y}', \mathbf{w}) \in \mathcal{R}\}$, we write $\epsilon(|\mathbf{x}|, |\mathbf{y}|, \delta)$ or $\epsilon(\delta)$ if the length is clear from the context.

It is well-known that IOPPs can be compiled into non-interactive arguments by the BCS compiler and Fiat-Shamir transform. However, in this case, standard soundness will not be sufficient. In fact, for Fiat-Shamir security, we need a stronger notion of round-by-round soundness. Before we formally define it, we first define a state function. Intuitively, it associates every (partial) transcript with a state that is either 0 or 1. A 0 state indicates that the verifier is going to reject the proof, while the verifier will accept the proof in state 1. A malicious prover starts with state 0, and succeeds in cheating if he ends up with state 1.

Definition 2.8 (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 on inputs $\mathbb{x}, \mathbb{y}$ and a transcript tr , outputs a bit, such that:*

- *Empty transcript:* if $\text{tr} = \epsilon$ is the empty string, then $\text{State}(\mathbb{x}, \mathbb{y}, \text{tr}) = 1$ 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 $\text{State}(\mathbb{x}, \mathbb{y}, \text{tr}) = 0$, then for every prover message π , $\text{State}(\mathbb{x}, \mathbb{y}, \text{tr}||\pi) = 0$.
- *Full transcript:* if tr is a full transcript and $\text{State}(\mathbb{x}, \mathbb{y}, \text{tr}) = 0$, then $\mathbf{V}$ rejects given this interaction transcript.
- *Default:* whenever tr is any transcript and $\text{State}(\mathbb{x}, \mathbb{y}, \text{tr}) = 1$, then every extension tr' of tr satisfies $\text{State}(\mathbb{x}, \mathbb{y}, \text{tr}') = 1$.

Informally, round-by-round soundness guarantees the probability of changing state 0 to state 1 in some round is small, and therefore with high probability a malicious prover would end up with state 0 and be rejected by the verifier. In particular, for IOPPs the probability of changing state from 0 to 1 can be large, as long as the instance $(\mathbb{x}, \mathbb{y})$ is sufficiently close to the language.

Definition 2.9 (Round-by-round Soundness [BCS16, CCH⁺19]). *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 $(\epsilon_1, \dots, \epsilon_k)$ if the IOPP has a state function State , such that for every $\mathbb{x}, \mathbb{y}$ and transcript $\text{tr} = (\gamma_1, \pi_1, \dots, \gamma_{i-1}, \pi_{i-1})$,*

$$\Pr_{\gamma_i}[\text{State}(\mathbb{x}, \mathbb{y}, \text{tr}) = 0 \wedge \text{State}(\mathbb{x}, \mathbb{y}, \text{tr}||\gamma_i) = 1] \leq \epsilon_i(\mathbb{x}, \mathbb{y}).$$

3 IOPP with Optimal Proximity Gaps

3.1 A Generic Folding Scheme

We first introduce a general folding framework for polynomial ideal codes [GSS00, Definition 1], [BHKS21, Definition 3.1]. The key observation is that the usual FRI folding operation can be viewed as a consequence of reducing a polynomial modulo a pair of suitably related factors and then applying the fold. This perspective extends naturally beyond Reed-Solomon codes.

To begin with, we introduce some related notions. It is not hard to see that every polynomial $f(x) \in \mathbb{F}^{<d}[x]$ can be uniquely decomposed as

$$f(x) = f_0(x^2) + x f_1(x^2),$$

where $f_0, f_1 \in \mathbb{F}^{<d/2}[x]$. We refer to f_0 and f_1 as the even and odd components of f , respectively.

Definition 3.1 (Fold). Let $f(x) = f_0(x^2) + xf_1(x^2) \in \mathbb{F}^{<d}[x]$. For $\gamma \in \mathbb{F}$, the fold of f with challenge γ is the polynomial

$$\text{Fold}(f, \gamma)(y) := f_0(y) + \gamma f_1(y) \in \mathbb{F}^{<\lfloor d/2 \rfloor}[y].$$

Thus, folding maps a polynomial in x to a polynomial in $y = x^2$ of approximately half the degree.

In the following, we study folding from a CRT viewpoint. Let $\vec{a} = (a_1, \dots, a_m) \in \mathbb{F}_q$ and denote by $e_{\vec{a}}(x)$ the product of linear factors $(x - a_i)$, i.e., $e_{\vec{a}}(x) = \prod_{i=1}^m (x - a_i)$. Similarly, we define $e_{-\vec{a}}(x) = \prod_{i=1}^m (x + a_i)$. Let

$$h_{\vec{a}}(x^2) := e_{\vec{a}}(x) \cdot e_{-\vec{a}}(x),$$

which is a polynomial in x^2 . Note that as long as all $a_i \neq 0$, $e_{\vec{a}}(x)$ and $e_{-\vec{a}}(x)$ are coprime. This allows a CRT reconstruction.

Lemma 3.2 (Bézout Coefficients). [vzGG13, Section 3.4] Suppose $e_{\vec{a}}(x), e_{-\vec{a}}(x) \in \mathbb{F}[x]$ are defined as above. Then there exist unique polynomials $u(x), v(x) \in \mathbb{F}[x]$ of degree less than m such that

$$u(x) \cdot e_{\vec{a}}(x) + v(x) \cdot e_{-\vec{a}}(x) = 1.$$

Moreover, if $\mathbb{F}_q$ is an FFT-friendly field, then u and v can be computed in $O(m \log^2 m)$ field operations via the extended Euclidean algorithm. In this work, we focus on FFT-friendly fields $\mathbb{F}_q$ that have sufficiently large 2-smooth multiplicative subgroups.

Lemma 3.3 (CRT Reconstruction). Let $e_{\vec{a}}(x), e_{-\vec{a}}(x), u(x)$, and $v(x)$ be defined as in [Definition 3.2](#). Assume

$$r_0(x) := f(x) \mod e_{\vec{a}}(x), \quad \text{and} \quad r_1(x) := f(x) \mod e_{-\vec{a}}(x).$$

Let $h_{\vec{a}}(x^2) := e_{\vec{a}}(x) \cdot e_{-\vec{a}}(x)$. The residual polynomial

$$g(x) := f(x) \mod h_{\vec{a}}(x^2) \in \mathbb{F}^{<2m}[x]$$

can be computed from (r_0, r_1) as follows:

$$g(x) \leftarrow v(x) \cdot e_{-\vec{a}}(x) \cdot r_0(x) + u(x) \cdot e_{\vec{a}}(x) \cdot r_1(x) \mod h_{\vec{a}}(x^2).$$

As shown in [Lemma 3.3](#) the polynomial $g(x)$ can be recovered from $r_0(x)$ and $r_1(x)$ via CRT. We refer to polynomial reduction modulo $h_{\vec{a}}(x^2)$ as *CRT reduction*. Now, we are ready to show our core observation.

Lemma 3.4 (Folding Commutes with CRT Reduction). Let $e_{\vec{a}}(x), e_{-\vec{a}}(x)$ be defined as above. Define $h_{\vec{a}}(x^2) := e_{\vec{a}}(x) \cdot e_{-\vec{a}}(x)$ and $g(x) = f(x) \mod h_{\vec{a}}(x^2)$. Then for every $\gamma \in \mathbb{F}$,

$$\text{Fold}(f, \gamma)(y) = \text{Fold}(g, \gamma)(y) \mod h_{\vec{a}}(y).$$

Proof. Recall $f(x) = f_0(x^2) + x \cdot f_1(x^2)$. By the definition of CRT Reconstruction, we have $f(x) = q(x) \cdot h_{\vec{a}}(x^2) + g(x)$ with $\deg(g(x)) < 2m$ and the pair $(q(x), g(x))$ is unique. Write $q(x), g(x)$ as

$$q(x) = q_0(x^2) + x \cdot q_1(x^2), \quad \text{and} \quad g(x) = g_0(x^2) + x \cdot g_1(x^2),$$

where $\deg(g_0) < m$ and $\deg(g_1) < m$. Comparing even and odd parts gives

$$f_0(y) = h_{\vec{a}}(y) \cdot q_0(y) + g_0(y), \quad \text{and} \quad f_1(y) = h_{\vec{a}}(y) \cdot q_1(y) + g_1(y).$$

Thus,

$$f_0(y) = g_0(y) \pmod{h_{\vec{a}}(y)}, \quad \text{and} \quad f_1(y) = g_1(y) \pmod{h_{\vec{a}}(y)}.$$

By definition, $\text{Fold}(f, \gamma)(y) := f_0(y) + \gamma \cdot f_1(y)$ and thus it holds that

$$\text{Fold}(f, \gamma) = f_0(y) + \gamma \cdot f_1(y) = g_0(y) + \gamma \cdot g_1(y) \pmod{h_{\vec{a}}(y)}.$$

In addition, from the definition of Fold , it holds that

$$\text{Fold}(r, \gamma) = r_0(y) + \gamma \cdot r_1(y).$$

This completes the proof. $\square$

Lemma 3.4 directly yields the commutative diagram:

$$\begin{array}{ccc} f(x) & \xrightarrow{\text{mod } h_{\vec{a}}(x^2)} & g(x) \\ \downarrow \text{Fold} & & \downarrow \text{Fold} \\ \text{Fold}(f, \gamma)(y) & \xrightarrow{\text{mod } h_{\vec{a}}(y)} & \text{Fold}(g, \gamma)(y) \end{array}$$

Thus folding and CRT reduction commute, enabling a generic IOPP for a code given by polynomials. In each round of the folding phase, after receiving the challenge from the verifier, the prover applies Fold and obtains the next round polynomial. While in the query phase, the verifier samples a random $\vec{a}$ and queries $f(x) \pmod{h_{\vec{a}}(x^2)}$ and $\text{Fold}(f, \gamma)(y) \pmod{h_{\vec{a}}(y)}$ respectively, and checks the equality of $\text{Fold}(g, \gamma)$ computed in the above two ways.

Example 3.5 (Reed-Solomon Codes). In the case of Reed-Solomon(RS) codes, $m = 1$ and we can set $e_a(x) = x - a$. In this way, each symbol of an RS codeword is given by $f(x) \pmod{e_a(x)}$, where a belongs to the RS evaluation set. Then $h_a(x^2) = x^2 - a^2$, $r_0 = f(a)$ and $r_1 = f(-a)$. Moreover, it follows that $u(x) = -1/(2a)$, $v(x) = 1/(2a)$ and

$$\begin{aligned} g(x) &= \frac{x+a}{2a} f(a) - \frac{x-a}{2a} f(-a) \pmod{h_a(x^2)} \\ &= \frac{x}{2a} (f(a) - f(-a)) + \frac{1}{2} (f(a) + f(-a)) \pmod{h_a(x^2)}. \end{aligned}$$

Given a challenge γ , $g(x)$ is folded into

$$\frac{\gamma}{2a} (f(a) - f(-a)) + \frac{1}{2} (f(a) - f(-a)),$$

which equals to $\text{Fold}(f, \gamma)(a^2)$. One can easily check that this coincides with the standard FRI folding procedure.

Example 3.6 (Folded RS Codes). For folded Reed-Solomon (FRS) codes, each symbol of a codeword corresponds to an evaluation vector $\vec{a} = (a, \zeta a, \dots, \zeta^{m-1}a)$, where ζ is typically a primitive element. In this case, we can set

$$e_{\vec{a}}(x) := \prod_{j=0}^{m-1} (x - \zeta^j a),$$

which splits completely over $\mathbb{F}$. This allows to view every codeword symbol as a polynomial $f(x) \bmod e_{\vec{a}}(x)$ and we can design an IOPP for FRS codes following the above commutative diagram. We refer details to [Section 3.2](#).

Example 3.7 (Univariate Multiplicity Codes). For univariate multiplicity (UM) codes, each symbol of a codeword is a vector

$$\left(f^{(0)}(a), f^{(1)}(a), \dots, f^{(m-1)}(a) \right),$$

where $f^{(j)}(x)$ is the j -th derivative of $f(x)$ and a belongs to the UM evaluation set. By Hermite interpolation, it is known that each symbol one-to-one corresponds to the polynomial

$$f(x) \bmod (x - a)^m.$$

This means we can set $e_a(x) := (x - a)^m$ and an IOPP for UM codes naturally follows from the above commutative diagram. We refer details to [Section 3.3](#).

3.2 IOPP for Folded Reed-Solomon Codes

We now specialize the general folding framework to FRS codes. Let $\zeta \in \mathbb{F}^*$ be an element of sufficiently large *odd* order. Let $\vec{a} = (a, \zeta a, \dots, \zeta^{m-1}a)$, and define $e_{\vec{a}}(x) := \prod_{j=0}^{m-1} (x - \zeta^j a)$. For convenience, write

$$e_{a,j}(x) := x - \zeta^j a \text{ and thus } e_{\vec{a}}(x) = \prod_{j=0}^{m-1} e_{a,j}(x).$$

I.e., $e_{\vec{a}}(x)$ splits completely over $\mathbb{F}$. Suppose $\alpha \in \mathbb{F}^*$ satisfies $\text{ord}(\alpha) = N = 2^n$ and consider the evaluation set $\mathcal{L} = \{\alpha^0, \alpha^1, \dots, \alpha^{N-1}\}$ and its associated cosets

$$\zeta^j \mathcal{L} := \{\zeta^j \cdot \alpha^i\}_{i \in [0, N-1]},$$

for $j \in [0, m-1]$. Assuming that all elements in $\zeta^j \mathcal{L}$ are distinct, an FRS codeword associated with a message polynomial $f(x) \in \mathbb{F}_q^{< RmN}[x]$ can be represented as the following $m \times N$ matrix:

$$\begin{pmatrix} f(\zeta^0 \alpha^0) & f(\zeta^0 \alpha^1) & \cdots & f(\zeta^0 \alpha^{N-1}) \\ f(\zeta^1 \alpha^0) & f(\zeta^1 \alpha^1) & \cdots & f(\zeta^1 \alpha^{N-1}) \\ \vdots & \vdots & \ddots & \vdots \\ f(\zeta^{m-1} \alpha^0) & f(\zeta^{m-1} \alpha^1) & \cdots & f(\zeta^{m-1} \alpha^{N-1}) \end{pmatrix}.$$

This is the standard point evaluation representation used in the list-decoding and proximity-gap analysis of FRS codes. In particular, we also view an FRS code, denoted by $\text{FRS}[\mathcal{L}, \zeta, R]$, as a function $\mathcal{L} \rightarrow \mathbb{F}_q^m$. To make our FRS code foldable across the folding phase, we choose α, ζ according to [Lemma 4.16](#) which guarantees that

$$\zeta^{j2^r} \mathcal{L}^{2^r} := \{\zeta^{j2^r} \cdot \alpha^{i2^r}\}_{i \in [0, m-1], i \in [0, \frac{N}{2^r}-1]}$$

induces an m -folded Reed-Solomon code $\text{FRS}[\mathcal{L}^{2^r}, \zeta^{2^r}, R]$ for any $r \leq n$. Moreover, we show in [Section 4.1](#) that such $\text{FRS}[\mathcal{L}^{2^r}, \zeta^{2^r}, R]$ still admit the optimal proximity gaps $1 - R - \varepsilon$.

Now we can feed FRS codes into the generic folding framework in [Section 3.1](#). However, the consistency check seems quite expensive, where the verifier needs to compute $g(x) := f(x) \bmod h_{\bar{a}}(x^2)$ from $r_0 = f(x) \bmod e_{\bar{a}}(x)$ and $r_1 = f(x) \bmod e_{-\bar{a}}(x)$ by CRT reconstruction. This step generally requires $O(m \log^2 m)$ field operations even if we assume an FFT-friendly field.

In the following, we show that it is unnecessary to explicitly reconstruct $g(x)$. Instead, it suffices to fold “row-by-row”. This is because $e_{\bar{a}}(x)$ splits completely over $\mathbb{F}_q$, which means that $f(x) \bmod e_{\bar{a}}(x)$ can be uniquely reconstructed from $\{f(x) \bmod e_{a,j}(x)\}_{j \in [0, m-1]}$ by CRT. Hence, $g(x)$ is uniquely determined by m linear polynomials $g_j(x) = f(x) \bmod h_{a,j}(x^2)$ where $j \in [0, m-1]$, and the equality check

$$\text{Fold}(f, \gamma)(y) \bmod h_{\bar{a}}(y) \stackrel{?}{=} \text{Fold}(g(x), \gamma)(y)$$

is equivalent to check $\text{Fold}(f, \gamma)(\zeta^{2^j} a^2) \stackrel{?}{=} g_j(\gamma)$ for all $j \in [0, m-1]$.

For completeness, we prove the equivalence by showing that folding commute with multi-CRT reduction.

Lemma 3.8 (Folding Commutes with Multi-CRT Reduction). *For $j \in [0, m-1]$, let $e_{a,j}(x) = x - \zeta^j a$, $e_{-a,j}(x) = x + \zeta^j a$ be coprime linear polynomials and define a polynomial $h_{a,j}(x^2)$ as $h_{a,j}(x^2) := e_{a,j}(x) \cdot e_{-a,j}(x)$. Assume*

$$r_{0,j} = f(x) \bmod e_{a,j}(x), \quad \text{and } r_{1,j} = f(x) \bmod e_{-a,j}(x).$$

Define $g_j(x) := f(x) \bmod h_{a,j}(x^2)$ as in [Definition 3.3](#). Then for every $\gamma \in \mathbb{F}$,

$$\text{Fold}(f, \gamma)(y) \equiv \begin{cases} \text{Fold}(g_0, \gamma)(y) & \bmod h_{a,0}(y), \\ \text{Fold}(g_1, \gamma)(y) & \bmod h_{a,1}(y), \\ \vdots & \vdots \\ \text{Fold}(g_{m-1}, \gamma)(y) & \bmod h_{a,m-1}(y). \end{cases}$$

In particular, $g_j(x)$ can be reconstructed from $r_{0,j}, r_{1,j}$ as in [Definition 3.3](#).

Proof. Write $f(x) = f_0(x^2) + x \cdot f_1(x^2)$. For $j \in [0, m-1]$, assume there exists $q_j(x), g_j(x)$ with $\deg(g_j) < 2$ such that $f(x) = q_j(x) \cdot h_{a,j}(x^2) + g_j(x)$. Since $h_{a,j}(x^2)$ is even, we may decompose $q_j(x), g_j(x)$ as

$$q_j(x) = q_{j,0}(x^2) + x \cdot q_{j,1}(x^2), \text{ and } g_j(x) = g_{j,0}(x^2) + x \cdot g_{j,1}(x^2),$$

where $\deg(r_{j,0}) < 1$ and $\deg(r_{j,1}) < 1$. Comparing the even and odd parts gives

$$f_0(y) = h_{a,j}(y) \cdot q_{j,0}(y) + g_{j,0}(y), \text{ and } f_1(y) = h_{a,j}(y) \cdot q_{j,1}(y) + g_{j,1}(y).$$

Consequently,

$$f_0(y) = g_{j,0}(y) \pmod{h_{a,j}(y)}, \text{ and } f_1(y) = g_{j,1}(y) \pmod{h_{a,j}(y)}.$$

By definition of the Fold operation,

$$\text{Fold}(f, \gamma)(y) := f_0(y) + \gamma \cdot f_1(y).$$

Therefore, for every $j \in [0, m-1]$,

$$\begin{aligned} \text{Fold}(f, \gamma)(y) &= f_0(y) + \gamma \cdot f_1(y) \\ &= g_{j,0}(y) + \gamma \cdot g_{j,1}(y) \pmod{h_{a,j}(y)} \\ &= \text{Fold}(g_j, \gamma)(y) \pmod{h_{a,j}(y)}. \end{aligned}$$

This completes the proof. □

Indeed, [Lemma 3.8](#) implies the following commutative diagram:

$$\begin{array}{ccc} f(x) & \xrightarrow{\text{mod } \{h_{a,j}(x^2)\}_{j \in [0, m-1]}} & \{g_j(x)\}_{j \in [0, m-1]} \\ \downarrow \text{Fold} & & \downarrow \text{Fold} \\ \text{Fold}(f, \gamma) & \xrightarrow{\text{mod } \{h_{a,j}(y)\}_{j \in [0, m-1]}} & \{g_j(\gamma)\}_{j \in [0, m-1]}, \end{array}$$

In summary, the IOPP for FRS codes have two phases: folding phase and query phase. In each round of the folding phase, the prover given the challenge γ , folds the codeword in a very similar way to FRI and sends the folded codeword as oracle message to the verifier. While in the query phase, for each pair of adjacent proofs $(f, \text{Fold}(f, \gamma))$, the verifier randomly picks a challenge $a = \alpha^i \in \mathcal{L}$, and queries

$$\{f(\zeta^j a), f(-\zeta^j a), \text{Fold}(f, \gamma)(\zeta^{2j} \cdot a^2)\}_{j \in [0, m-1]}.$$

For each row $j \in [0, m-1]$, the verifier checks the following equation

$$\text{Fold}(f, \gamma)(\zeta^{2j} \cdot a^2) = \frac{\gamma}{2\zeta^j \cdot a} (f(\zeta^j \cdot a) - f(-\zeta^j \cdot a)) + \frac{1}{2} (f(\zeta^j \cdot a) + f(-\zeta^j \cdot a)), \quad (4)$$

or equivalently,

$$\text{Fold}(f, \gamma) (\zeta^{2j} a^2) \stackrel{?}{=} g_j(\gamma).$$

Note that for each row j , the consistency check only depends on the queries on this row, avoiding cross-row computations. In particular, it is easy to see that checking [Eq.\(4\)](#) only takes $O(1)$ field operations, thus checking m rows need only $O(m)$ time. In addition, recall that after folding, the new FRS code is defined over $\zeta^{2j}\mathcal{L} := \{\zeta^{2j}\alpha^{2i} : i \in [0, N/2 - 1]\}$ for each $j \in [0, m - 1]$. Hence the prover can also use [Eq.\(4\)](#) to compute the folded codeword, which can be done in time $O(mN)$.

Fast Encoding Algorithm of FRS codes. Finally, we give a simple, fast encoding algorithm for FRS codes via the perspective of CRT. Let $f(x)$ be a polynomial of degree at most RmN . For an FRS codeword $f \in \text{FRS}[\mathcal{L}, \zeta, R]$, we can define the j -th row polynomial:

$$f_j(x) := f(x) \mod (x^{2^n} - \zeta^{j \cdot 2^n}),$$

such that $f_j(\zeta^j \alpha_i) := f(\zeta^j \alpha_i)$ for each $i \in [0, 2^n - 1]$, and $j \in [0, m - 1]$. We also define the shifted j -th row polynomial

$$h_j(x) = f(\zeta^j x) \mod (x^{2^n} - 1).$$

Algorithm 1 Fast Encoding of FRS

Input: Polynomial $f(x)$ over $\mathbb{F}_q$ of degree at most RmN , and FRS evaluation set $\mathcal{L} = \{\zeta^j \alpha_i \mid j \in [0, m - 1], i \in [0, 2^n - 1]\}$.

Output: An FRS codeword $f \in \text{FRS}[\mathcal{L}, \zeta, R]$.

- 1: **for** $j \in [0, m - 1]$ **do**
 - 2: Compute $h_j(x) = f(\zeta^j x) \mod (x^{2^n} - 1)$.
 - 3: Run FFT on the polynomial $h_j(x)$ with respect to the evaluation set $\{\alpha_1, \dots, \alpha_{2^n}\}$, and obtain $\vec{f}_j$.
 - 4: **end for**
 - 5: Interpolate $\vec{f}_0, \dots, \vec{f}_{m-1}$ as $f \in \text{FRS}[\mathcal{L}, \zeta, R]$ and return f .
-

Since $\mathcal{L} = \{\alpha_1, \dots, \alpha_{2^n}\}$ forms a multiplicative subgroup of $\mathbb{F}_q^*$ of order 2^n , they are essentially the roots of $x^{2^n} - 1 = 0$ over $\mathbb{F}_q$. It is clear that

$$h_j(\alpha^i) = f(\zeta^j \alpha^i).$$

Thus, to evaluate f at m rows, it suffices to evaluate m shifted polynomials $h_0(x), \dots, h_{m-1}(x)$ at evaluation set $\{\alpha_1, \dots, \alpha_{2^n}\}$. Again since $\mathcal{L}$ is 2-smooth, we can employ FFT for fast evaluation. The resulting FRS encoding time is $O(mN \log N)$, where $N = 2^n$. From above discussions, we are able to extend FRI to IOPP for FRS codes. A self-contained construction is given as follows.

Procedure 1: $\pi_{\text{IOPP-FRS}}$

Let $\mathbb{F}_q$ be a field, $\zeta \in \mathbb{F}_q$ be a primitive element, and $\alpha \in \mathbb{F}_q$ has order $N = 2^n$. Let $\mathcal{L} = \{\alpha^i \mid i \in [0, N-1]\}$. Let $\text{FRS}[\mathcal{L}, \zeta, R]$ be a Folded Reed-Solomon code over $\mathbb{F}_q$ with rate $R \in (0, 1)$.

1. $\mathcal{P}$ sends to $\mathcal{V}$ an oracle message $f_0 : \mathcal{L} \rightarrow \mathbb{F}_q^m$, which should be close to a codeword $f_0 \in \text{FRS}[\mathcal{L}, \zeta, R]$.
2. **Folding phase:** for each round $i = 1, \dots, k$, where $k \leq n$, proceed as follows:
 - (a) $\mathcal{V}$ samples $\gamma_i \xleftarrow{\$} \mathbb{F}_q$, and sends it to $\mathcal{P}$.
 - (b) $\mathcal{P}$ computes $f_i \leftarrow \text{Fold}(f_{i-1}, \gamma_i)$, and sends an oracle message $f_i : \mathcal{L}^{2^i} \rightarrow \mathbb{F}_q^m$ to $\mathcal{V}$, which should be close to a codeword $f_i \in \text{FRS}[\mathcal{L}^{2^i}, \zeta^{2^i}, R]$. If $i = k$, $\mathcal{P}$ sends f_i to $\mathcal{V}$ in the clear.
 - (c) If $i = k$, $\mathcal{V}$ checks whether $f_k \in \text{FRS}[\mathcal{L}^{2^n}, \zeta^{2^n}, R]$.
3. **Query phase:** for each repetition $t = 1, \dots, \ell$, $\mathcal{V}$ proceeds as follows:
 - (a) Sample a random value $s_t \xleftarrow{\$} \{\alpha^0, \alpha^1, \dots, \alpha^{N-1}\}$, and set $s_t^{(0)} = s_t$.
 - (b) Query f_0 at the symbol indexed by $\pm s_t^{(0)}$.
 - (c) For each $i = 1, \dots, k$:
 - Set $s_t^{(i)} := (s_t^{(i-1)})^2$.
 - Query f_i at the symbol indexed by $\pm s_t^{(i)}$. Namely, a set of evaluations $\{f_i(\zeta^{2^i j} s_t^{(i)}), f_i(-\zeta^{2^i j} s_t^{(i)})\}_{j \in [0, m-1]}$.
 - Check consistency of f_{i-1} at $\pm s_t^{(i-1)}$ and f_i at $s_t^{(i)}$ according to [Eq.\(4\)](#).
4. $\mathcal{V}$ accepts if and only if all the above checks pass.

Theorem 3.9. *IOPP-FRS is a secure IOPP for FRS codes with $O(mN)$ prover time, $O(\lambda \log N)$ query complexity, $O(\lambda m \log N)$ verifier time.*

Proof. Note that in round i of folding phase, the prover performs $O(mN/2^i)$ field operations, hence $\sum_{i=0}^{k-1} O(mN/2^i) = O(mN)$ in total. In query phase, the consistency check would be repeated $\ell = O(\lambda)$ times, and each consists of $O(\log N)$ queries, hence $O(\lambda \log N)$ in total. Finally, the verifier time is $O(\lambda m \log N)$ field operations due to $O(\lambda)$ repetitions for checking consistency of $O(\log N)$ oracles.

We establish soundness following the round-by-round(RBR) analysis [BGK⁺23, GMW25] of FRI. Suppose that f_0 is δ -far from $\text{FRS}[\mathcal{L}, \zeta, R]$ and $\text{State}(f_0, \gamma_0) = 0$, in the first round there are two strategies for a malicious prover to fool the verifier:

- (1) “luck out” in the sense the challenge $\gamma_0 \xleftarrow{\$} \mathbb{F}_q$ sampled by the verifier leads to an $f_1 = \text{Fold}(f_0, \gamma_0)$ that is δ -close to $\text{FRS}[\mathcal{L}^2, \zeta^2, R]$; or (2) send $f'_1 \neq f_1$ that is δ -close to $\text{FRS}[\mathcal{L}^2, \zeta^2, R]$. In fact, strategy (2) does not give any advantage to the prover, because intuitively f'_1 will differ from f_1 in many positions and this will be detected by the consistency check in the query phase. Thus in the folding phase, the prover’s optimal strategy is to behave honestly and hope to “luck out”. We can generalize the above argument to all rounds of the folding phase. In more detail, suppose $\text{State}(f_0, \gamma_0, f'_1, \gamma_1, \dots, f'_{i-1}, \gamma_{i-1}) = 0$, the state changes to 1 in the next round (i.e., $\text{State}(f_0, \gamma_0, f'_1, \gamma_1, \dots, f'_i, \gamma_i) = 1$) if and only if (1) the prover “lucks out” and f_i is δ -close to $\text{FRS}[\mathcal{L}^{2^i}, \zeta^{2^i}, R]$ and (2) all $f'_j = f_j$ for

$j = 1, \dots, i$. Now it suffices to bound the probability that the prover “lucks out”. We claim that any challenge γ_i that makes the prover “luck out” in round i is a bad challenge in mutual correlated argument (MCA, see [Definition 2.6](#)). Thus the RBR soundness in folding phase is determined by the error function of MCA, denoted by

$$\epsilon^{\text{MCA}}(\text{FRS}[\mathcal{L}^{2^i}, \zeta^{2^i}, R], \delta).$$

The above claim is proved in [Lemma 3.10](#).

To complete the RBR soundness, it remains to consider the querying phase. Recall that the verifier will sample ℓ values from $\{\alpha, \alpha^2, \dots, \alpha^N\}$, query $f_0, \dots, f_k$ at corresponding positions, and perform ℓ parallel consistency checks. We note that the state changes from 0 to 1 only if all the consistency check pass. Thus the RBR soundness in query phase is determined by $(\epsilon^{\text{qry}})^\ell$, where ϵ^{qry} denotes the probability that a single consistency check passes. Following the same argument in [\[GMW25\]](#), we conclude that $\epsilon^{\text{qry}} \leq 1 - \delta$.

Combining together, the RBR soundness is upper bounded by

$$\max\{\epsilon^{\text{MCA}}(\text{FRS}[\mathcal{L}^{2^i}, \zeta^{2^i}, R], \delta), (1 - \delta)^\ell\}.$$

□

Lemma 3.10 (Folding preserves agreement set). *Consider FRS codes $\text{FRS}[\mathcal{L}, \zeta, R]$ and $\text{FRS}[\mathcal{L}^2, \zeta^2, R]$, where $\mathcal{L}^2 = \{\alpha^2 \mid \alpha \in \mathcal{L}\}$. Suppose $\text{FRS}[\mathcal{L}^2, \zeta^2, R]$ has $(2, \delta, \epsilon)$ -MCA. Then for every function $f : \mathcal{L} \rightarrow \mathbb{F}_q^m$,*

$$\Pr_{\gamma \in \mathbb{F}_q} \left[\begin{array}{l} |S_1| \geq (1 - \delta)N/2, \\ \exists S_0 \subseteq [\mathcal{L}] \text{ s.t. } \wedge \exists c' \in \text{FRS}[\mathcal{L}^2, \zeta^2, R], \text{ such that } \text{Fold}(f, \gamma)|_{S_1} = c'|_{S_1}, \\ \wedge \forall c \in \text{FRS}[\mathcal{L}, \zeta, R] \text{ such that } c|_{S_0} \neq f|_{S_0} \end{array} \right] \leq \epsilon,$$

where $S_1 := \{\alpha^2 \mid \alpha \in S_0\}$.

Proof. Note that every $f \in \text{FRS}[\mathcal{L}, \zeta, R]$ determines $f_0, f_1 \in \text{FRS}[\mathcal{L}^2, \zeta^2, R]$ such that $f(a) = f_0(a^2) + a \cdot f_1(a^2)$ for any $a \in \mathcal{L}$. By our choices of $\zeta^j \mathcal{L}, \zeta^{2j} \mathcal{L}^2$, we have that for any $a^2 \in \zeta^{2j} \mathcal{L}^2$,

$$f_0(a^2) = \frac{f(a) + f(-a)}{2}, \quad f_1(a^2) = \frac{f(a) - f(-a)}{2a}.$$

Thus for arbitrary functions $f : \mathcal{L} \rightarrow \mathbb{F}_q^m$, we can similarly define functions $f_0, f_1 : \mathcal{L}^2 \rightarrow \mathbb{F}_q^m$. Note that $\text{Fold}(f, \gamma) = f_0(x) + \gamma \cdot f_1(x)$. Hence we have

$$\begin{aligned} & \Pr_{\gamma \in \mathbb{F}_q} \left[\begin{array}{l} |S_1| \geq (1 - \delta)N/2, \\ \exists S_0 \subseteq [\mathcal{L}] \text{ s.t. } \wedge \exists c' \in \text{FRS}[\mathcal{L}^2, \zeta^2, R], \text{ such that } \text{Fold}(f, \gamma)|_{S_1} = c'|_{S_1}, \\ \wedge \forall c \in \text{FRS}[\mathcal{L}, \zeta, R] \text{ such that } c|_{S_0} \neq f|_{S_0} \end{array} \right] \\ & \leq \Pr_{\gamma \in \mathbb{F}_q} \left[\begin{array}{l} |S_1| \geq (1 - \delta)N/2, \\ \exists S_0 \subseteq [\mathcal{L}] \text{ s.t. } \wedge \exists c' \in \text{FRS}[\mathcal{L}^2, \zeta^2, R], \text{ such that } (f_0 + \gamma \cdot f_1)|_{S_1} = c'|_{S_1}, \\ \wedge \forall c \in \text{FRS}[\mathcal{L}, \zeta, R] \text{ such that } c|_{S_0} \neq f|_{S_0} \end{array} \right]. \end{aligned}$$

Now the above probability differs from MCA only in the last condition, i.e.,

$$\forall c \in \text{FRS}[\mathcal{L}, \zeta, R] \text{ such that } c|_{S_0} \neq f|_{S_0}, \quad (5)$$

while in [Definition 2.6](#), it is

$$\exists i \in [0, 1] \forall c' \in \text{FRS}[\mathcal{L}^2, \zeta^2, R] \text{ such that } c'|_{S_1} \neq f_i|_{S_1}, \quad (6)$$

Thus, to complete the proof, it suffices to show that [Eq.\(5\)](#) implies [Eq.\(6\)](#). We prove it by contradiction. Assume that [Eq.\(6\)](#) does not hold, then for every f_0, f_1 , there exist $c'_0, c'_1 \in \text{FRS}[\mathcal{L}^2, \zeta^2, R]$ that agree with f_0, f_1 on S_1 . We claim that there will exist a $c \in \text{FRS}[\mathcal{L}, \zeta, R]$ that agrees with f on S_0 , which contradicts with [Eq.\(5\)](#).

To prove the above claim, we first define a bivariate polynomial

$$Q(X, Y) = \sum_{\alpha^2 \in \tilde{S}} \sum_{j=0}^{m-1} I_{\zeta^{2j}\alpha^2}(X) \cdot (f_0(\zeta^{2j}\alpha^2) + Y \cdot f_1(\zeta^{2j}\alpha^2)),$$

where $\tilde{S} \subseteq S_1$ has size $RN/2$, and $I_a(z)$ is an indicator function, i.e., $I_a(z) = 1$ if $z = a$ and 0 otherwise. Clearly, $Q(X, Y)$ has degree less than $RmN/2$ on the variable X and degree 1 on the variable Y . Moreover, it is clear that for all $\alpha^2 \in \tilde{S}$, and $j \in [0, m-1]$

$$Q(\zeta^{2j}\alpha^2, Y) = f_0(\zeta^{2j}\alpha^2) + Y \cdot f_1(\zeta^{2j}\alpha^2) = c'_0(\zeta^{2j}\alpha^2) + Y \cdot c'_1(\zeta^{2j}\alpha^2) \quad (7)$$

In particular, $Q(X, Y)$ can be seen as a univariate polynomial on the variable X of degree less than $RmN/2$. Similarly, $c'_0(X) + c'_1(X) \cdot Y$ is also a univariate polynomial on the variable X of degree less than $RmN/2$. Since $Q(X, Y)$ and $c'_0(X) + c'_1(X) \cdot Y$ agree on $\tilde{S}$ and its cosets, they must be equal. That is,

$$Q(X, Y) = c'_0(X) + c'_1(X) \cdot Y.$$

This immediately implies that for all $\alpha^2 \in S_1$, and $j \in [0, m-1]$

$$Q(\zeta^{2j}\alpha^2, Y) = c'_0(\zeta^{2j}\alpha^2) + c'_1(\zeta^{2j}\alpha^2) \cdot Y.$$

Finally, we can define $c(X) = Q(X^2, X)$, whose degree is at most

$$2(RmN/2 - 1) + 1 = RmN - 1 < RmN.$$

Hence $c \in \text{FRS}[\mathcal{L}, \zeta, R]$. In addition, c agrees with f on S_0 and its cosets because for every $\alpha \in S_0$, and $j \in [0, m-1]$

$$\begin{aligned} c(\zeta^j\alpha) &= Q(\zeta^{2j}\alpha^2, \zeta^j\alpha) = c'_0(\zeta^{2j}\alpha^2) + \zeta^j\alpha \cdot c'_1(\zeta^{2j}\alpha^2) \\ &= f_0(\zeta^{2j}\alpha^2) + \zeta^j\alpha \cdot f_1(\zeta^{2j}\alpha^2) = f(\zeta^j\alpha) \end{aligned}$$

This completes the claim as well as the proof. $\square$

3.3 IOPP of Univariate Multiplicity Codes

As discussed in [Example 3.7](#), for univariate multiplicity codes, we can define

$$e_a(x) := (x - a)^m \text{ and } e_{-a}(x) := (x + a)^m,$$

and we can design IOPP for UM codes in a similar way to [Section 3.2](#). The remaining thing is efficiency. Basically, we need to compute $g(x) := f(x) \bmod (x^2 - a^2)^m$ from $r_0(x) := f(x) \bmod (x - a)^m$ and $r_1(x) := f(x) \bmod (x + a)^m$ efficiently. By [Lemma 3.3](#), it suffices to find $u(x), v(x)$ of degree less than m such that

$$u(x)e_a(x) + v(x)e_{-a}(x) = 1.$$

By the generic extended Euclidean algorithm, $u(x), v(x)$ can be computed in $O(m \log^2 m)$ time, if the underlying field is FFT-friendly. We demonstrate below that for the specific pair

$$((x - a)^m, (x + a)^m),$$

the polynomial pair $(u(x), v(x))$ can be computed in $O(m \log m)$ time, thus saving a $\log m$ multiplicative factor.

We first recall two standard facts regarding polynomial derivatives and formal power series. Suppose $\mathbb{F}_q$ is an FFT friendly field.

Lemma 3.11 (Taylor Shift). *[vzGG13, Section 9.2] Given a polynomial $P(x) = \sum_{i=0}^{n-1} p_i x^i \in \mathbb{F}^{<n}[x]$, and $c \in \mathbb{F}$, the shifted polynomial $Q(y) = P(x + c)$ can be computed in $O(n \log n)$ time.*

Lemma 3.12 (Power-series for Square Root and Inverse). *[vzGG13, Section 9.1] Let $\mathbb{F}[[x]]$ denote the ring of formal power series over $\mathbb{F}$. Given $P(x) \in \mathbb{F}[[x]]$ with $P(0) \neq 0$, both truncated series*

$$\sqrt{P(x)} \bmod (x^n) \text{ and } P^{-1}(x) \bmod (x^n)$$

can be computed in $O(n \log n)$ time.

Corollary 3.13. *Given $e_a(x) := (x - a)^m$ and $e_{-a}(x) := (x + a)^m$, the unique polynomials $(u(x), v(x)) \in \mathbb{F}^{<m}[x]$ satisfying*

$$u(x)e_a(x) + v(x)e_{-a}(x) = 1 \tag{8}$$

can be computed in $O(m \log m)$ time.

Proof. [Eq.\(8\)](#) is equivalent to the pair of congruences

$$u(x) \equiv e_a(x)^{-1} \bmod e_{-a}(x) \text{ and } v(x) \equiv e_{-a}(x)^{-1} \bmod e_a(x).$$

Equivalently, this requires computing

$$(x - a)^{-m} \bmod (x + a)^m \text{ and } (x + a)^{-m} \bmod (x - a)^m.$$

Let $y = x + a$. The first term transforms to $(y - 2a)^{-m} \bmod y^m$. Since $2a \neq 0$, the constant term $-2a$ is non-zero. Thus, by [Lemma 3.12](#), the truncated inverse series modulo y^m can be computed in $O(m \log m)$ time. Applying a Taylor shift $y \mapsto x + a$ yields $u(x)$ in $O(m \log m)$ time. Similarly, $v(x)$ is computed in $O(m \log m)$ time. Consequently, $u(x)$ and $v(x)$ are computed in $O(m \log m)$ total time. $\square$

Now given

$$r_0(x) = f(x) \bmod (x - a)^m, \quad \text{and } r_1(x) = f(x) \bmod (x + a)^m.$$

By [Lemma 3.3](#), the reconstruction of $g(x)$ is given by

$$g(x) := v(x)e_{-a}(x)r_0(x) + u(x)e_a(x)r_1(x) \bmod (x^2 - a^2)^m.$$

Since $\mathbb{F}_q$ is FFT-friendly, computing polynomial multiplications needs $O(m \log m)$ time. In addition, as $\deg((x^2 - a^2)^m) = 2m$ and $v(x), u(x), e_a(x), e_{-a}(x), r_0(x), r_1(x)$ all are polynomials of degree less than m , modulo $(x^2 - a^2)^m$ can be done in $O(m \log m)$ time. Hence, reconstruction of $g(x)$ takes $O(m \log m)$ time.

Now following the commutative diagram in [Section 3.1](#), we give an IOPP for UM codes as follows.

Procedure 2: $\pi_{\text{IOPP-UM}}$

Let $\mathbb{F}_q$ be a field, and $\alpha \in \mathbb{F}_q$ has order $N = 2^n$. Let $\mathcal{L} = \{\alpha^i : i \in [0, N - 1]\}$ and $\text{UM}[\mathcal{L}, R]$ be a univariate multiplicity code over $\mathbb{F}_q$ with rate $R \in (0, 1)$.

1. $\mathcal{P}$ sends to $\mathcal{V}$ an oracle message $f_0 : \mathcal{L} \rightarrow \mathbb{F}_q^m$, which should be close to a codeword $f_0 \in \text{UM}[\mathcal{L}, R]$.
2. For each round $i = 1, \dots, k$, where $k \leq n$, $\mathcal{P}$ and $\mathcal{V}$ proceed as follows:
 - (a) $\mathcal{V}$ samples $\gamma_i \xleftarrow{\$} \mathbb{F}_q$, and sends it to $\mathcal{P}$.
 - (b) $\mathcal{P}$ computes $f_i \leftarrow \text{Fold}(f_{i-1}, \gamma_i)$, and sends an oracle message $f_i : \mathcal{L}^{2^i} \rightarrow \mathbb{F}_q^m$ to $\mathcal{V}$, which should be close to a codeword $f_i \in \text{UM}[\mathcal{L}^{2^i}, R]$. If $i = k$, $\mathcal{P}$ sends f_i to $\mathcal{V}$ in the clear.
 - (c) If $i = k$, $\mathcal{V}$ checks whether $f_k \in \text{UM}[\mathcal{L}^{2^k}, R]$.
3. For each repetition $t = 1, \dots, \ell$, $\mathcal{V}$ proceeds as follows:
 - (a) Sample a random value $s_t \xleftarrow{\$} \mathcal{L}$, and set $s_t^{(0)} = s_t$.
 - (b) Query f_0 at the symbol indexed by $\pm s_t^{(0)}$, i.e., $r_{0,0}(x) := f_0(x) \bmod (x - s_t^{(0)})^m$, and $r_{0,1}(x) := f_0(x) \bmod (x + s_t^{(0)})^m$.
 - (c) For each round $i = 1, \dots, k$:
 - Reconstruct $g_{i-1}(x) := f_{i-1}(x) \bmod (x^2 - (s_t^{(i-1)})^2)^m$ from $r_{i-1,0}(x)$ and $r_{i-1,1}(x)$.
 - Set $s_t^{(i)} := (s_t^{(i-1)})^2$.
 - Query f_i at the symbol indexed by $\pm s_t^{(i)}$, i.e., $r_{i,0}(x) := f_i(x) \bmod (x - s_t^{(i)})^m$, and $r_{i,1}(x) := f_i(x) \bmod (x + s_t^{(i)})^m$.
 - Check $r_{i,0}(x) \stackrel{?}{=} \text{Fold}(g_{i-1}, \gamma_i)$.
4. $\mathcal{V}$ accepts if and only if all the above checks pass.

Theorem 3.14. *IOPP-UM is a secure IOPP for UM codes with $O(Nm \log m)$ prover time, $O(\lambda \log N)$ query complexity, $O(\lambda m \log m \log N)$ verifier time.*

Proof. The proof follows similarly as the proof of [Theorem 3.9](#).

3.4 Concrete Proof Size

In this section, we show the details to compute the proof size for our IOPPs for m -folded FRS codes and UM codes after applying the BCS compiler [\[BCS16\]](#).

Let $N = 2^s$, $k = N \cdot k_0$ and $n = N \cdot m$. Assume the Merkle tree hash output is of $\text{hash-output-bit} = 256$ -bit. The code rate is $R = k/n = k_0/m$.

Assume the code polynomial is folded $r \leq s$ times. The commitment has r oracles, i.e., $(f_0, f_1 \dots f_{r-1})$ and the last polynomial has $N \cdot k_0 / 2^r = 2^{s-r} \cdot k_0$ coefficients. Then the commitment size is $r \cdot \text{hash-output-bit} + 2^{s-r} \cdot k_0 \cdot \text{field-size}$.

Assume the consistency check repeats t times.

- f_0 opening has $2m$ field elements and $s - 1$ hash outputs.
- f_1 opening has $2m$ field elements and $s - 2$ hash outputs.
- f_{r-1} opening has $2m$ field elements and $s - (r - 1 + 1) = s - r$ hash outputs.
- f_r outputs nothing.

Hence, the consistency checks outputs $2mrt$ field elements and $t(sr - (r + 1)r/2)$ hash output elements, i.e., $2mrt \cdot \text{field-size} + t \cdot (sr - (r + 1)r/2) \cdot \text{hash-output-bit}$. In summary, the total proof size is

$$(r + tsr - t(r + 1)r/2) \cdot \text{hash-output-bit} + (2mrt + 2^{s-r} \cdot k_0) \cdot \text{field-size}.$$

Table 3: Real FRS or UM IOPP Proof Size with $\lambda = 100$

R	k	m	ϵ	t	field-size	proof-size	soundness	r	R	k	m	ϵ	t	field-size	proof-size	soundness	r
1/2	2^{16}	32	0.2885	292	128	904.41	118.16	1	1/4	2^{16}	16	0.3475	135	128	482.47	114.31	3
	2^{18}	32	0.2885	292	128	1716.59	116.22	3		2^{18}	16	0.3475	135	128	739.88	112.31	5
	2^{20}	32	0.2885	292	128	2565.28	114.24	5		2^{20}	16	0.3475	135	128	1014.16	110.31	7
	2^{22}	32	0.2885	292	128	3450.47	112.24	7		2^{22}	16	0.3475	135	128	1305.31	108.31	9
	2^{24}	32	0.2885	292	128	4372.16	110.24	9		2^{24}	16	0.3475	135	128	1613.34	106.31	11
	2^{26}	32	0.2885	292	128	5330.34	108.24	11		2^{26}	16	0.3475	135	128	1938.25	104.31	13
	2^{28}	32	0.2885	292	128	6325.03	106.24	13		2^{28}	16	0.3475	135	128	2280.03	102.31	15
	2^{30}	32	0.2885	292	128	7356.22	104.24	15		2^{30}	16	0.3475	135	128	2638.69	100.31	17
R	k	m	ϵ	t	field-size	proof-size	soundness	r	R	k	m	ϵ	t	field-size	proof-size	soundness	r
1/8	2^{16}	16	0.2948	80	128	345.59	112.19	3	1/16	2^{16}	16	0.2492	60	128	285.38	110.92	4
	2^{18}	16	0.2948	80	128	503.16	110.20	5		2^{18}	16	0.2492	60	128	407.31	108.95	6
	2^{20}	16	0.2948	80	128	670.72	108.20	7		2^{20}	16	0.2492	60	128	536.75	106.95	8
	2^{22}	16	0.2948	80	128	848.28	106.20	9		2^{22}	16	0.2492	60	128	673.69	104.95	10
	2^{24}	16	0.2948	80	128	1035.84	104.20	11		2^{24}	16	0.2492	60	128	818.12	102.95	12
	2^{26}	16	0.2948	80	128	1233.41	102.20	13		2^{26}	16	0.2492	60	128	970.06	100.95	14
	2^{28}	16	0.2948	80	128	1440.97	100.20	15		2^{28}	16	0.2492	60	192	1401.50	162.95	16
	2^{30}	16	0.2948	80	192	2054.06	162.20	18		2^{30}	16	0.2492	60	192	1598.44	160.95	18

The proof sizes of our IOPPs for FRS and UM codes and of FRI are reported in [Table 3](#) and [Table 4](#), respectively. The FRI IOPP soundness is computed using the recent progress of [\[BCH⁺26\]](#). To account for 64-bit machine word granularity, the field representation sizes are rounded up to multiples of 64 bits. For typical parameter settings as show in [Table 3](#), our IOPPs achieve 100-bit security with a

128-bit field representation, whereas FRI requires a 192-bit representation. For FRS/UM codes we use optimal proximity gaps $1 - R - \epsilon$ while for RS codes we use proximity radius $1 - \sqrt{R} - \epsilon$.

Table 5 provides conjectured parameter choices satisfying $m = O(1/\epsilon)$ that would yield smaller IOPP proofs than those of FRI in Table 4. We leave to future work the challenge of reducing the parameter m from $O(1/\epsilon^2)$ to $O(1/\epsilon)$ while retaining the optimal proximity gap for FRS and UM codes.

Table 4: FRI Proof Size with $\lambda = 100$

R	k	ϵ	t	field-size	proof-size	soundness	r	R	k	ϵ	t	field-size	proof-size	soundness	r
1/2	2^{16}	0.00002	203	192	476.75	103.54	4	1/4	2^{16}	0.00002	102	192	279.25	103.54	5
	2^{18}	0.00002	203	192	705.19	101.54	6		2^{18}	0.00002	102	192	394.06	101.54	7
	2^{20}	0.00003	203	192	959.00	102.46	8		2^{20}	0.00003	102	192	521.62	102.46	9
	2^{22}	0.00004	203	192	1238.19	102.54	10		2^{22}	0.00004	102	192	661.94	102.54	11
	2^{24}	0.00005	203	192	1542.75	102.14	12		2^{24}	0.00005	102	192	815.00	102.15	13
	2^{26}	0.00006	203	192	1872.69	101.46	14		2^{26}	0.00006	102	192	980.81	101.46	15
	2^{28}	0.00008	203	192	2228.00	101.53	16		2^{28}	0.00008	102	192	1159.38	101.54	17
	2^{30}	0.00010	203	192	2608.69	101.14	18		2^{30}	0.00010	102	192	1350.69	101.14	19
1/8	2^{16}	0.00002	68	192	202.22	103.54	5	1/16	2^{16}	0.00002	51	192	158.06	103.54	6
	2^{18}	0.00002	68	192	278.78	101.54	7		2^{18}	0.00002	51	192	215.50	101.54	8
	2^{20}	0.00003	68	192	363.84	102.46	9		2^{20}	0.00003	51	192	279.31	102.46	10
	2^{22}	0.00004	68	192	457.41	102.53	11		2^{22}	0.00004	51	192	349.50	102.54	12
	2^{24}	0.00005	68	192	559.47	102.14	13		2^{24}	0.00005	51	192	426.06	102.14	14
	2^{26}	0.00006	68	192	670.03	101.46	15		2^{26}	0.00006	51	192	509.00	101.46	16
	2^{28}	0.00008	68	192	789.09	101.53	17		2^{28}	0.00008	51	192	598.31	101.53	18
	2^{30}	0.00010	68	192	916.66	101.14	19		2^{30}	0.00010	51	192	694.00	101.14	20

Table 5: Conjectured FRS or UM IOPP Proof Size with $\lambda = 100$

R	k	m	ϵ	t	field-size	proof-size	soundness	r	R	k	m	ϵ	t	field-size	proof-size	soundness	r
1/2	2^{16}	4	0.2100	203	128	375.50	117.55	3	1/4	2^{16}	4	0.1750	82	128	198.72	114.40	5
	2^{18}	4	0.2100	203	128	572.22	115.72	5		2^{18}	4	0.1750	82	128	283.34	112.69	7
	2^{20}	4	0.2100	203	128	794.31	113.77	7		2^{20}	4	0.1750	82	128	378.22	110.77	9
	2^{22}	4	0.2100	203	128	1041.78	111.78	9		2^{22}	4	0.1750	82	128	483.34	108.79	11
	2^{24}	4	0.2100	203	128	1314.62	109.78	11		2^{24}	4	0.1750	82	128	598.72	106.80	13
	2^{26}	4	0.2100	203	128	1612.84	107.79	13		2^{26}	4	0.1750	82	128	724.34	104.80	15
	2^{28}	4	0.2100	203	128	1936.44	105.79	15		2^{28}	4	0.1750	82	128	860.22	102.80	17
	2^{30}	4	0.2100	203	128	2285.41	103.79	17		2^{30}	4	0.1750	82	128	1006.34	100.80	19
1/8	2^{16}	4	0.1400	53	128	148.09	112.22	5	1/16	2^{16}	4	0.1110	40	128	124.94	103.68	6
	2^{18}	4	0.1400	53	128	206.12	110.67	7		2^{18}	4	0.1110	40	128	171.25	103.65	8
	2^{20}	4	0.1400	53	128	270.78	108.81	9		2^{20}	4	0.1110	40	128	222.56	103.57	10
	2^{22}	4	0.1400	53	128	342.06	106.85	11		2^{22}	4	0.1110	40	128	278.88	103.27	12
	2^{24}	4	0.1400	53	128	419.97	104.86	13		2^{24}	4	0.1110	40	128	340.19	102.47	14
	2^{26}	4	0.1400	53	128	504.50	102.86	15		2^{26}	4	0.1110	40	128	406.50	101.03	16
	2^{28}	4	0.1400	53	128	595.66	100.86	17		2^{28}	4	0.1110	40	192	530.81	163.21	18
	2^{30}	4	0.1400	53	192	769.94	162.86	20		2^{30}	4	0.1110	40	192	612.12	161.26	20

4 Sharp Proximity Gaps for FRS Codes and UM Codes

The argument proceeds in four stages. First, the subspace design property shows all codewords close to a curve inside a low dimensional affine space. Second, an pruning procedure identifies a common set of coordinates and yields a weak curve decoding guarantee. Third, list decodability amplifies this guarantee so that sufficiently many nearby codewords lie on one codeword curve. Finally,

interpolation converts curve decodability into correlated and mutual correlated agreement, from which the desired line proximity gap follows by [Lemma 4.13](#). Throughout the argument, we retain the exact finite parameter losses needed for the final instantiation. For a curve $\mathbf{u}_\alpha = \sum_{j=0}^\ell \mathbf{u}_j \alpha^j$, the possible symbols at coordinate i lie in the affine space $u_{0,i} + \text{span}_{\mathbb{F}_q} \{u_{1,i}, \dots, u_{\ell,i}\}$. Our first step is therefore to control codewords that are close to a product of affine spaces.

Lemma 4.1. *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be a τ subspace design code. For each $i \in [N]$, let $Y_i = \mathbf{y}_i + L_i \subseteq \mathbb{F}_q^m$ be an affine $\mathbb{F}_q$ subspace, where $L_i \subseteq \mathbb{F}_q^m$ is a linear subspace. Define $Y_\delta^{\text{aff}} := \{\mathbf{c} \in \mathcal{C} : \Delta(\mathbf{c}, Y_1 \times \dots \times Y_N) \leq \delta\}$ where $\Delta(\mathbf{c}, Y_1 \times \dots \times Y_N) := \frac{1}{N} |\{i \in [N] : \mathbf{c}_i \notin Y_i\}|$. Assume that $Y_\delta^{\text{aff}} \neq \emptyset$ and fix any base point $\mathbf{c}^{(0)} \in Y_\delta^{\text{aff}}$. Define space $H_\delta := \text{span}_{\mathbb{F}_q} \{\mathbf{c} - \mathbf{c}^{(0)} : \mathbf{c} \in Y_\delta^{\text{aff}}\} \subseteq \mathcal{C}$. If $\delta < 1 - \frac{s\tau(s)+1}{s+1} - \frac{1}{N(s+1)} \sum_{i=1}^N \dim_{\mathbb{F}_q}(L_i)$, then $\dim_{\mathbb{F}_q}(H_\delta) < s$.*

Proof. Suppose, toward a contradiction, that $\dim_{\mathbb{F}_q}(H_\delta) \geq s$. Since H_δ is spanned by the differences $\mathbf{c} - \mathbf{c}^{(0)}$ with $\mathbf{c} \in Y_\delta^{\text{aff}}$, we can choose $\mathbf{c}^{(1)}, \dots, \mathbf{c}^{(s)} \in Y_\delta^{\text{aff}}$ such that $\mathbf{c}^{(1)} - \mathbf{c}^{(0)}, \dots, \mathbf{c}^{(s)} - \mathbf{c}^{(0)}$ are linearly independent over $\mathbb{F}_q$. Let $A := \text{span}_{\mathbb{F}_q} \{\mathbf{c}^{(1)} - \mathbf{c}^{(0)}, \dots, \mathbf{c}^{(s)} - \mathbf{c}^{(0)}\} \subseteq \mathcal{C}$. Then $\dim_{\mathbb{F}_q} A = s$. For each $i \in [N]$, define $A_i := \{\mathbf{a} \in A : \mathbf{a}_i = \mathbf{0}\}$ and $B_i := \{j \in \{0, 1, \dots, s\} : \mathbf{c}_i^{(j)} \in Y_i\}$. Since each $\mathbf{c}^{(j)}$ belongs to Y_δ^{aff} , we have $\sum_{i=1}^N |B_i| \geq (s+1)(1-\delta)N$.

Fix $i \in [N]$, and define $Q_i := \{\mathbf{a} \in A : \mathbf{a}_i \in L_i\} \subseteq A$. We claim that $|B_i| - 1 \leq \dim_{\mathbb{F}_q}(Q_i)$. If $B_i = \emptyset$, there is nothing to prove. Otherwise, fix some $j_0 \in B_i$. For every $j \in B_i \setminus \{j_0\}$, we have $\mathbf{c}_i^{(j)}, \mathbf{c}_i^{(j_0)} \in Y_i = \mathbf{y}_i + L_i$ and hence $(\mathbf{c}^{(j)} - \mathbf{c}^{(j_0)})_i \in L_i$. Thus, $\mathbf{c}^{(j)} - \mathbf{c}^{(j_0)} \in Q_i$ for every $j \in B_i \setminus \{j_0\}$. Moreover, the vectors $\mathbf{c}^{(j)} - \mathbf{c}^{(j_0)}$ ($j \in B_i \setminus \{j_0\}$) are linearly independent over $\mathbb{F}_q$. Indeed, $\mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \dots, \mathbf{c}^{(s)}$ are affinely independent, because $\mathbf{c}^{(1)} - \mathbf{c}^{(0)}, \dots, \mathbf{c}^{(s)} - \mathbf{c}^{(0)}$ are linearly independent. Therefore, any collection of differences from a fixed base point among these affinely independent points is linearly independent. Hence $|B_i| - 1 \leq \dim_{\mathbb{F}_q}(Q_i)$.

Now consider the coordinate projection $Q_i \rightarrow L_i$, $\mathbf{a} \mapsto \mathbf{a}_i$. Its kernel is exactly A_i , and its image is contained in L_i . Therefore $\dim_{\mathbb{F}_q}(Q_i) \leq \dim_{\mathbb{F}_q}(A_i) + \dim_{\mathbb{F}_q}(L_i)$. Combining this with the previous inequality gives $|B_i| \leq 1 + \dim_{\mathbb{F}_q}(A_i) + \dim_{\mathbb{F}_q}(L_i)$. Summing over all $i \in [N]$, we obtain $(s+1)(1-\delta)N \leq \sum_{i=1}^N |B_i| \leq N + \sum_{i=1}^N \dim_{\mathbb{F}_q}(A_i) + \sum_{i=1}^N \dim_{\mathbb{F}_q}(L_i)$.

Since $\dim_{\mathbb{F}_q} A = s$, the τ subspace design property of $\mathcal{C}$ gives $\sum_{i=1}^N \dim_{\mathbb{F}_q}(A_i) \leq N \cdot \tau(s) \cdot s$. Therefore, $(s+1)(1-\delta)N \leq N + N \cdot \tau(s) \cdot s + \sum_{i=1}^N \dim_{\mathbb{F}_q}(L_i)$. Rearranging, we get $\delta \geq 1 - \frac{s\tau(s)+1}{s+1} - \frac{1}{N(s+1)} \sum_{i=1}^N \dim_{\mathbb{F}_q}(L_i)$, contradicting the hypothesis. Hence, $\dim_{\mathbb{F}_q}(H_\delta) < s$.

[Lemma 4.1](#) shows that all relevant codewords differ from a fixed base point by vectors in a low dimensional linear space H . We next prune H while retaining the exact success probability $\varepsilon/(\rho \dim_{\mathbb{F}_q} H + \varepsilon)$ needed for the final parameter bound. For a nonzero subspace $H \subseteq \mathcal{C}$, define $w_i(H) := \mathbf{1}_{H_i \neq H}(\rho \dim_{\mathbb{F}_q}(H_i) + \varepsilon)$ and let $D_{\rho, \varepsilon}(H)$ be the distribution on $[N]$ given by $\Pr_{i \sim D_{\rho, \varepsilon}(H)}[i = j] := \frac{w_j(H)}{\sum_{t=1}^N w_t(H)}$.

Theorem 4.2. Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be a τ subspace design code. Let $H \subseteq \mathcal{C}$ be an $\mathbb{F}_q$ linear subspace with $\dim_{\mathbb{F}_q} H < s$. Let $\rho \in [0, 1]$ and $\varepsilon > 0$ satisfy $\rho = 1 - \tau(s) - \varepsilon$. Let $\mathbf{y} \in (\mathbb{F}_q^m)^N$, and fix any $\mathbf{c} \in H$ satisfying $\Delta(\mathbf{c}, \mathbf{y}) \leq \rho$. Then the set S output by $\text{PRUNED}_{\rho, \varepsilon}(H)$ satisfies $\Pr_S[\mathbf{c}_i = \mathbf{y}_i \text{ for every } i \in S \text{ and } H_S = \{\mathbf{0}\}] \geq \frac{\varepsilon}{\rho \dim_{\mathbb{F}_q} H + \varepsilon} \geq \frac{\varepsilon}{\rho(s-1) + \varepsilon}$.

Algorithm 2 $\text{PRUNED}_{\rho, \varepsilon}(H)$

```

1: if  $H = \{\mathbf{0}\}$  then
2:   return  $\emptyset$ .
3: end if
4: Sample  $i \in [N]$  according to  $D_{\rho, \varepsilon}(H)$ .
5: return  $\{i\} \cup \text{PRUNED}_{\rho, \varepsilon}(H_i)$ .

```

Proof. The procedure is well defined and terminates. Indeed, if a recursive state $H \neq \{\mathbf{0}\}$, then not all coordinates satisfy $H_i = H$; otherwise every element of H is zero in every coordinate, and hence $H = \{\mathbf{0}\}$. Moreover, whenever $H_i \neq H$, we have $H_i \subsetneq H$, and therefore $\dim_{\mathbb{F}_q}(H_i) < \dim_{\mathbb{F}_q} H$. Thus, the recursion terminates. If the output is S , then $H_S = \{\mathbf{0}\}$.

Define $f_{\rho, \varepsilon}(H) := \frac{1}{\rho \dim_{\mathbb{F}_q} H + \varepsilon}$ for every $\mathbb{F}_q$ linear subspace $H \subseteq \mathcal{C}$. For $S \subseteq [N]$, define the indicator variable $X_S = 1$ if $\mathbf{c}_i = \mathbf{y}_i, \forall i \in S$ and 0 otherwise. We first prove the following one step inequality. Let $S \subseteq [N]$ satisfy $H_S \neq \{\mathbf{0}\}$. Then

$$\mathbb{E}_{i \sim D_{\rho, \varepsilon}(H_S)} [X_{S \cup \{i\}} \cdot f_{\rho, \varepsilon}(H_{S \cup \{i\}})] \geq X_S \cdot f_{\rho, \varepsilon}(H_S).$$

Let $I := \{i \in [N] : H_{S \cup \{i\}} = H_S\}$. If $X_S = 0$, then the inequality is immediate. Hence we assume $X_S = 1$. Since $\Delta(\mathbf{c}, \mathbf{y}) \leq \rho = 1 - \tau(s) - \varepsilon$, we have $|\{i \in [N] : \mathbf{c}_i = \mathbf{y}_i\}| \geq (\tau(s) + \varepsilon)N$. Therefore, $|\{i \notin I : \mathbf{c}_i = \mathbf{y}_i\}| \geq (\tau(s) + \varepsilon)N - |I|$.

Since $H_S \subseteq H$, we have $\dim_{\mathbb{F}_q}(H_S) \leq \dim_{\mathbb{F}_q} H < s$. Thus, applying the τ subspace design property of $\mathcal{C}$ to H_S , we obtain $\sum_{i=1}^N \dim_{\mathbb{F}_q}(H_{S \cup \{i\}}) \leq N \cdot \tau(s) \cdot \dim_{\mathbb{F}_q}(H_S)$. Now, using the definition of the distribution $D_{\rho, \varepsilon}(H_S)$, we obtain

$$\begin{aligned}
\mathbb{E}_{i \sim D_{\rho, \varepsilon}(H_S)} [X_{S \cup \{i\}} \cdot f_{\rho, \varepsilon}(H_{S \cup \{i\}})] &= \sum_{i=1}^N \Pr_{i \sim D_{\rho, \varepsilon}(H_S)}[i = i] \cdot X_{S \cup \{i\}} \cdot f_{\rho, \varepsilon}(H_{S \cup \{i\}}) \\
&= \sum_{\substack{i \notin I \\ \mathbf{c}_i = \mathbf{y}_i}} \frac{\rho \dim_{\mathbb{F}_q}(H_{S \cup \{i\}}) + \varepsilon}{\sum_{j=1}^N w_j(H_S)} \cdot \frac{1}{\rho \dim_{\mathbb{F}_q}(H_{S \cup \{i\}}) + \varepsilon} \\
&= \frac{|\{i \notin I : \mathbf{c}_i = \mathbf{y}_i\}|}{\sum_{j=1}^N w_j(H_S)} \\
&\geq \frac{(\tau(s) + \varepsilon)N - |I|}{\rho(N \cdot \tau(s) \cdot \dim_{\mathbb{F}_q}(H_S) - |I| \dim_{\mathbb{F}_q}(H_S)) + \varepsilon(N - |I|)}.
\end{aligned}$$

We claim that the last expression is at least $\frac{1}{\rho \dim_{\mathbb{F}_q}(H_S) + \varepsilon}$. After cross multiplying, the difference between the left hand side and the right hand side is

$$\begin{aligned}
&((\tau(s) + \varepsilon)N - |I|)(\rho \dim_{\mathbb{F}_q}(H_S) + \varepsilon) - (\rho(N \tau(s) \dim_{\mathbb{F}_q}(H_S) - |I| \dim_{\mathbb{F}_q}(H_S)) + \varepsilon(N - |I|)) \\
&= \varepsilon N(\rho \dim_{\mathbb{F}_q}(H_S) + \tau(s) + \varepsilon - 1) = \varepsilon \rho N(\dim_{\mathbb{F}_q}(H_S) - 1) \geq 0,
\end{aligned}$$

where we used $\rho = 1 - \tau - \varepsilon$ and $\dim_{\mathbb{F}_q} H_S \geq 1$, since $H_S \neq \{0\}$. Therefore

$$\mathbb{E}_{i \sim D_{\rho, \varepsilon}(H_S)} [X_{S \cup \{i\}} \cdot f_{\rho, \varepsilon}(H_{S \cup \{i\}})] \geq X_S \cdot f_{\rho, \varepsilon} H_S.$$

Starting from $S = \emptyset$, we have $X_\emptyset \cdot f_{\rho, \varepsilon}(H) = \frac{1}{\rho \dim_{\mathbb{F}_q} H + \varepsilon}$. Iterating the one-step inequality along the pruning process gives $\mathbb{E}_{S \sim \text{PRUNED}_{\rho, \varepsilon}(H)} [X_S \cdot f_{\rho, \varepsilon}(H_S)] \geq \frac{1}{\rho \dim_{\mathbb{F}_q} H + \varepsilon}$. For the output set S , we always have $H_S = \{0\}$, and hence $f_{\rho, \varepsilon}(H_S) = \frac{1}{\varepsilon}$. Thus,

$$\Pr_S[\mathbf{c}_i = \mathbf{y}_i \text{ for every } i \in S \text{ and } H_S = \{0\}] = \mathbb{E}[X_S] \geq \frac{\varepsilon}{\rho \dim_{\mathbb{F}_q} H + \varepsilon}.$$

As $\dim_{\mathbb{F}_q} H < s$, this is at least $\frac{\varepsilon}{\rho(s-1) + \varepsilon}$.

Definition 4.3 ($\mathbb{F}_q$ **curve decodability** [GG26]). Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be an $\mathbb{F}_q$ linear code. We say that $\mathcal{C}$ is $\mathbb{F}_q$ -(ℓ, δ, a, b) curve decodable if, for every $\mathbf{u}_0, \dots, \mathbf{u}_\ell \in (\mathbb{F}_q^m)^N$ and every function $f : \mathbb{F}_q \rightarrow \mathcal{C}$, the set $A := \{\alpha \in \mathbb{F}_q : \Delta(\mathbf{u}_\alpha, f(\alpha)) \leq \delta\}$ having size at least a implies that there exist $\mathbf{c}_0, \dots, \mathbf{c}_\ell \in \mathcal{C}$ such that $f(\alpha) = \sum_{j=0}^\ell \mathbf{c}_j \alpha^j$ for at least b values $\alpha \in A$. When $\ell = 1$, we call this $\mathbb{F}_q$ line decodability.

For any $\alpha_1, \dots, \alpha_k$ in a field $\mathbb{F}$, the $k \times k$ Vandermonde matrix $V(\alpha_1, \dots, \alpha_k)$ is given by $V(\alpha_1, \dots, \alpha_k)_{i,j} = \alpha_i^{j-1}$. For all pairwise distinct $\alpha_1, \dots, \alpha_k$, the matrix $V(\alpha_1, \dots, \alpha_k)$ is invertible. Given $\beta \in \mathbb{F}_q$, define $\mathbf{m}_{\alpha_1, \dots, \alpha_k}(\beta)$ as

$$\mathbf{m}_{\alpha_1, \dots, \alpha_k}(\beta) V(\alpha_1, \dots, \alpha_k) = [1 \ \beta \ \dots \ \beta^{k-1}].$$

Lemma 4.4 ([GG26]). Let V be a linear space over $\mathbb{F}$. Let $\alpha_1, \dots, \alpha_k \in \mathbb{F}$ be distinct elements, and let $\mathbf{v}_{\alpha_1}, \dots, \mathbf{v}_{\alpha_k}$ be elements of V . Then there exist vectors $\mathbf{w}_0, \dots, \mathbf{w}_{k-1} \in V$ such that $\mathbf{v}_{\alpha_i} = \sum_{j=0}^{k-1} \mathbf{w}_j \cdot \alpha_i^j$ for every $i \in [k]$. Additionally, for any $\beta \in \mathbb{F}$, we have

$$\mathbf{m}_{\alpha_1, \dots, \alpha_k}(\beta) \begin{bmatrix} \mathbf{v}_{\alpha_1} \\ \mathbf{v}_{\alpha_2} \\ \vdots \\ \mathbf{v}_{\alpha_k} \end{bmatrix} = \sum_{j=0}^{k-1} \mathbf{w}_j \cdot \beta^j.$$

We now combine the affine dimension bound in Lemma 4.1 with pruning in Theorem 4.2 to obtain a weak curve decodability statement.

Theorem 4.5. Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be a τ subspace design code. Fix integers $\ell \geq 1$ and $1 \leq s \leq m$. Let $\rho \in [0, 1]$ satisfy $\rho < 1 - \tau(s) - \frac{\ell+1-\tau(s)}{s+1}$ and let $\varepsilon := 1 - \tau(s) - \rho$. Then, for every positive integer a , the code $\mathcal{C}$ is $\mathbb{F}_q$ -($\ell, \rho, a, \lceil \frac{\varepsilon a}{\rho(s-1) + \varepsilon} \rceil$) curve decodable.

Proof. Fix $\mathbf{u}_0, \dots, \mathbf{u}_\ell \in (\mathbb{F}_q^m)^N$ and a function $f : \mathbb{F}_q \rightarrow \mathcal{C}$. For each coordinate $i \in [N]$, define $L_i := \langle (\mathbf{u}_1)_i, \dots, (\mathbf{u}_\ell)_i \rangle_{\mathbb{F}_q} \subseteq \mathbb{F}_q^m$ and $Y_i := (\mathbf{u}_0)_i + L_i$. Then $\dim_{\mathbb{F}_q}(L_i) \leq \ell$ for every $i \in [N]$. Let

$$A := \left\{ \alpha \in \mathbb{F}_q : \Delta \left(\sum_{j=0}^{\ell} \mathbf{u}_j \cdot \alpha^j, f(\alpha) \right) \leq \rho \right\}$$

and assume $|A| \geq a$. For every $\alpha \in A$, the codeword $f(\alpha)$ belongs to $Y_\rho^{\text{aff}} := \{ \mathbf{c} \in \mathcal{C} : \Delta(\mathbf{c}, Y_1 \times \dots \times Y_N) \leq \rho \}$ because whenever $f(\alpha)_i = \left(\sum_{j=0}^{\ell} \mathbf{u}_j \cdot \alpha^j \right)_i$, this common value lies in Y_i .

Fix some $\alpha_* \in A$, and let $\mathbf{c}^{(*)} := f(\alpha_*)$. Define the space

$$H := \text{span}_{\mathbb{F}_q} \left\{ \mathbf{c} - \mathbf{c}^{(*)} : \mathbf{c} \in Y_\rho^{\text{aff}} \right\} \subseteq \mathcal{C}.$$

Since $\dim_{\mathbb{F}_q}(L_i) \leq \ell$ for all $i \in [N]$, we have

$$1 - \frac{s\tau(s) + 1}{s + 1} - \frac{1}{N(s + 1)} \sum_{i=1}^N \dim_{\mathbb{F}_q}(L_i) \geq 1 - \frac{s\tau(s) + 1}{s + 1} - \frac{\ell}{s + 1} = 1 - \tau(s) - \frac{\ell + 1 - \tau(s)}{s + 1}.$$

Thus, by the assumption on ρ , [Lemma 4.1](#) implies that $\dim_{\mathbb{F}_q} H < s$. Apply the pruning procedure of [Theorem 4.2](#) to the space H . For each fixed $\alpha \in A$, define $\mathbf{c}_\alpha := f(\alpha) - \mathbf{c}^{(*)} \in H$ and $\mathbf{y}_\alpha := \sum_{j=0}^{\ell} \mathbf{u}_j \cdot \alpha^j - \mathbf{c}^{(*)}$. Then $\Delta(\mathbf{c}_\alpha, \mathbf{y}_\alpha) = \Delta\left(f(\alpha), \sum_{j=0}^{\ell} \mathbf{u}_j \cdot \alpha^j\right) \leq \rho = 1 - \tau(s) - \varepsilon$. Therefore, by the pruning [Theorem 4.2](#),

$$\Pr_S \left[f(\alpha)_i = \left(\sum_{j=0}^{\ell} \mathbf{u}_j \cdot \alpha^j \right)_i \text{ for every } i \in S \text{ and } H_S = \{0\} \right] \geq \frac{\varepsilon}{\rho(s-1) + \varepsilon}.$$

Taking expectation over $\alpha \in A$, there exists a set $S \subseteq [N]$ such that $H_S = \{0\}$ and $B := \left\{ \alpha \in A : f(\alpha)|_S = \left(\sum_{j=0}^{\ell} \mathbf{u}_j \cdot \alpha^j \right)|_S \right\}$ satisfies $|B| \geq \left\lceil \frac{\varepsilon|A|}{\rho(s-1) + \varepsilon} \right\rceil \geq \left\lceil \frac{\varepsilon a}{\rho(s-1) + \varepsilon} \right\rceil$.

It remains to prove that the codewords $f(\alpha)$ for $\alpha \in B$ lie on one degree ℓ codeword curve. Write $B = \{\alpha_1, \dots, \alpha_b\}$ and let $d := \min\{b-1, \ell\}$. For any $\beta \in B$, by interpolation over $\{\alpha_1, \dots, \alpha_{d+1}\} \subseteq \mathbb{F}_q$, we have

$$\begin{aligned} & \left(\mathbf{m}_{\alpha_1, \dots, \alpha_{d+1}}(\beta) \begin{pmatrix} f(\alpha_1) \\ \vdots \\ f(\alpha_{d+1}) \end{pmatrix} - f(\beta) \right) \Big|_S \\ &= \left(\mathbf{m}_{\alpha_1, \dots, \alpha_{d+1}}(\beta) \begin{pmatrix} f(\alpha_1) - \mathbf{c}^{(*)} \\ \vdots \\ f(\alpha_{d+1}) - \mathbf{c}^{(*)} \end{pmatrix} - (f(\beta) - \mathbf{c}^{(*)}) \right) \Big|_S = 0. \end{aligned}$$

The vector inside the parentheses belongs to H , since it is an $\mathbb{F}_q$ linear combination of vectors of the form $f(\alpha) - \mathbf{c}^{(*)}$ with $f(\alpha) \in Y_\rho^{\text{aff}}$. Since its restriction to S is zero and $H_S = \{0\}$, it must be the zero codeword. Hence

$$\mathbf{m}_{\alpha_1, \dots, \alpha_{d+1}}(\beta) \begin{pmatrix} f(\alpha_1) \\ \vdots \\ f(\alpha_{d+1}) \end{pmatrix} = f(\beta)$$

for every $\beta \in B$.

By [Lemma 4.4](#), there exist $\mathbf{c}_0, \dots, \mathbf{c}_d \in \mathcal{C}$ such that $f(\beta) = \sum_{j=0}^d \mathbf{c}_j \cdot \beta^j$ for every $\beta \in B$. Setting $\mathbf{c}_{d+1} = \dots = \mathbf{c}_\ell = 0$ gives a degree at most ℓ codeword curve through all points $f(\beta)$, $\beta \in B$. This proves the theorem. $\square$

The conversion from weak curve decodability to correlated agreement follows three results from [\[GG26\]](#). First, list decodability amplifies the number of nearby parameters captured by a single codeword curve. Second, agreement with that curve on sufficiently many parameters bounds the total disagreement set of the coefficient words. Finally, this yields correlated agreement.

Theorem 4.6 ([\[GG26\]](#)). *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be an $\mathbb{F}_q$ linear code, let $\ell, t_1 > \ell$, $t_2 \in \mathbb{N}$, and $\delta \in (0, 1)$. Suppose that $\mathcal{C}$ is $\left(\delta \left(1 + \frac{\ell}{t_1 - \ell}\right), L\right)$ list decodable and $\mathbb{F}_q$ -(ℓ, δ, a, t_1) curve decodable. Then $\mathcal{C}$ is $\mathbb{F}_q$ -($\ell, \delta, (t_2 - 1)L + a, t_2$) curve decodable, provided that $q > \binom{L+1}{2}\ell$.*

Lemma 4.7 ([\[GG26\]](#)). *Let $\mathbf{u}_0, \dots, \mathbf{u}_\ell, \mathbf{c}_0, \dots, \mathbf{c}_\ell \in (\mathbb{F}_q^m)^N$, and let $B \subseteq \mathbb{F}_q$ with $|B| \geq t > \ell$. If $\Delta\left(\sum_{j=0}^\ell \mathbf{u}_j \alpha^j, \sum_{j=0}^\ell \mathbf{c}_j \alpha^j\right) \leq \delta$ for every $\alpha \in B$, then*

$$\left| \bigcup_{j=0}^\ell \text{supp}(\mathbf{u}_j - \mathbf{c}_j) \right| \leq \frac{\delta N}{1 - \ell/t}.$$

Applying [Lemma 4.7](#) to the resulting codeword curve gives the standard conversion from curve decodability to correlated agreement.

Theorem 4.8 ([\[GG26\]](#)). *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be an $\mathbb{F}_q$ linear code. Let $\ell, t > \ell$, $a \in \mathbb{N}$ and $\delta \in (0, 1)$. Suppose that $\mathcal{C}$ is $\mathbb{F}_q$ -(ℓ, δ, a, t) curve decodable for some $t > \ell$. Then $\mathcal{C}$ has $\mathbb{F}_q$ -($\ell, \delta, \frac{a}{q}, \frac{\ell}{t}$) curve correlated agreement.*

For mutual correlated agreement, we use a more precise counting argument. Keeping the exact integral threshold $\lfloor \ell \delta N + \ell \rfloor + 1$ is important, since this quantity appears directly in the final soundness bound.

Theorem 4.9. *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be an $\mathbb{F}_q$ linear code. Suppose that $\mathcal{C}$ is $\mathbb{F}_q$ -($\ell, \delta, a, \lfloor \ell \delta N + \ell \rfloor + 1$) curve decodable. Then $\mathcal{C}$ has $\mathbb{F}_q$ -($\ell, \delta, \frac{a}{q}$) mutual correlated agreement.*

Proof. Suppose not. Then there exist $\mathbf{u}_0, \dots, \mathbf{u}_\ell \in (\mathbb{F}_q^m)^N$ and some $\delta' \leq \delta$ such that the bad event in the definition of $\mathbb{F}_q$ mutual correlated agreement occurs for more than a values of $\alpha \in \mathbb{F}_q$. Let $A \subseteq \mathbb{F}_q$ be a set of such values with $|A| \geq a$.

For every $\alpha \in A$, choose a codeword $\mathbf{c}_\alpha \in \mathcal{C}$ and a maximal set $S_\alpha \subseteq [N]$ such that $|S_\alpha| \geq (1 - \delta')N$ and $\mathbf{c}_\alpha|_{S_\alpha} = \left(\sum_{j=0}^{\ell} \mathbf{u}_j \cdot \alpha^j \right)|_{S_\alpha}$. Moreover, for this S_α , there is some $j \in \{0, \dots, \ell\}$ such that $\mathbf{c}'|_{S_\alpha} \neq \mathbf{u}_j|_{S_\alpha}$ for all $\mathbf{c}' \in \mathcal{C}$. Define $f : \mathbb{F}_q \rightarrow \mathcal{C}$ by setting $f(\alpha) = \mathbf{c}_\alpha$ for $\alpha \in A$, and defining f arbitrarily outside A .

By the $\mathbb{F}_q$ -($\ell, \delta, a, \lfloor \ell\delta N + \ell \rfloor + 1$) curve decodability of $\mathcal{C}$, there exist $\mathbf{c}^{(0)}, \dots, \mathbf{c}^{(\ell)} \in \mathcal{C}$ and a subset $B \subseteq A$ with $|B| \geq \lfloor \ell\delta N + \ell \rfloor + 1$ such that $\mathbf{c}_\alpha = \sum_{j=0}^{\ell} \mathbf{c}^{(j)} \cdot \alpha^j$ for every $\alpha \in B$. Let

$$S := \left\{ i \in [N] : (\mathbf{c}^{(j)})_i = (\mathbf{u}_j)_i \text{ for every } j = 0, \dots, \ell \right\}.$$

For every $\alpha \in B$, we have $S \subseteq S_\alpha$ by the maximality of S_α . For $\alpha \in B$, write $E_\alpha := S_\alpha \setminus S$. Fix $i \notin S$. If $i \in E_\alpha$, then $\sum_{j=0}^{\ell} ((\mathbf{u}_j)_i - (\mathbf{c}^{(j)})_i) \cdot \alpha^j = 0$. Since $i \notin S$, the coefficients in the above expression are not all zero. By Vandermonde invertibility over $\mathbb{F}_q$, this can happen for at most ℓ values of $\alpha \in \mathbb{F}_q$. Hence, $\sum_{\alpha \in B} |E_\alpha| \leq \ell(N - |S|)$.

On the other hand, for every $\alpha \in B$, $\Delta \left(\sum_{j=0}^{\ell} \mathbf{u}_j \cdot \alpha^j, \sum_{j=0}^{\ell} \mathbf{c}^{(j)} \cdot \alpha^j \right) \leq \delta'$. Applying the interpolation counting lemma with the set B , we obtain $N - |S| \leq \frac{\delta' N}{1 - \ell/|B|} \leq \frac{\delta N}{1 - \ell/|B|}$. Thus, $\sum_{\alpha \in B} |E_\alpha| \leq \frac{\ell \delta N}{1 - \ell/|B|}$. Since $|B| \geq \lfloor \ell\delta N + \ell \rfloor + 1 > \ell\delta N + \ell$, we have $\frac{\ell \delta N}{1 - \ell/|B|} < |B|$. Therefore, at least one $\alpha \in B$ satisfies $E_\alpha = \emptyset$. For this α , we have $S_\alpha = S$. By the definition of S , for every $j = 0, \dots, \ell$, $\mathbf{c}^{(j)}|_{S_\alpha} = \mathbf{u}_j|_{S_\alpha}$. This contradicts the bad event for this α , because that event requires the existence of some $j_\alpha \in \{0, \dots, \ell\}$ such that $\mathbf{c}'|_{S_\alpha} \neq \mathbf{u}_{j_\alpha}|_{S_\alpha}$ for every $\mathbf{c}' \in \mathcal{C}$. Indeed, taking $\mathbf{c}' = \mathbf{c}^{(j_\alpha)} \in \mathcal{C}$ gives a contradiction. Therefore the bad event has probability at most a/q and the theorem follows. $\square$

Corollary 4.10. *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be $\mathbb{F}_q$ linear, with $\ell \geq 1$, $t_1 > \ell$, $t_2 \in \mathbb{N}$, and $\delta \in (0, 1)$. Assume $\mathcal{C}$ is $\left(\delta \left(1 + \frac{\ell}{t_1 - \ell} \right), L \right)$ list decodable and $\mathbb{F}_q$ -(ℓ, δ, a, t_1) curve decodable. For $q > \binom{L+1}{2}\ell$, $\mathcal{C}$ has $\mathbb{F}_q$ -($\ell, \delta, \frac{(t_2-1)L+a}{q}, \frac{\ell}{t_2}$) curve correlated agreement and $\mathbb{F}_q$ -($\ell, \delta, \frac{\lfloor \ell\delta N + \ell \rfloor L + a}{q}$) mutual correlated agreement.*

Proof. By [Theorem 4.6](#), the two assumptions imply that $\mathcal{C}$ is $\mathbb{F}_q$ -($\ell, \delta, (t_2 - 1)L + a, t_2$) curve decodable. Applying [Theorem 4.8](#) gives $\mathbb{F}_q$ -($\ell, \delta, \frac{(t_2-1)L+a}{q}, \frac{\ell}{t_2}$) curve correlated agreement.

For mutual correlated agreement, apply [Theorem 4.6](#) with $t_2 = \lfloor \ell\delta N + \ell \rfloor + 1$. This gives $\mathbb{F}_q$ -($\ell, \delta, \lfloor \ell\delta N + \ell \rfloor L + a, \lfloor \ell\delta N + \ell \rfloor + 1$) curve decodability. Then [Theorem 4.9](#) implies $\mathbb{F}_q$ -($\ell, \delta, \frac{\lfloor \ell\delta N + \ell \rfloor L + a}{q}$) mutual correlated agreement. $\square$

To instantiate the generic results above, we use the known subspace design bound of [\[GK16\]](#) and the list decoding bound of [\[CZ25\]](#) for FRS and UM codes.

Theorem 4.11 ([[GK16](#)]). *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be either an m FRS code or an order m UM code of rate R . In the multiplicity code case, assume that the message degree bound $k = mRN$ satisfies $k < \text{char}(\mathbb{F}_q)$. Then $\mathcal{C}$ is a τ subspace design code for $\tau(s) = \frac{mR}{m-s+1}$ for all $1 \leq s \leq m$, and $\tau(s) = 1$ otherwise.*

Besides the subspace design property, the final theorem requires a list decodability input up to the capacity radius. We record the list decoding bounds for FRS codes and UM codes below.

Theorem 4.12 ([[CZ25](#)]). *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be either an m FRS code or an order m UM code of rate R . In the multiplicity code case, assume that $q = p$ is prime. Then, for every integer L with $1 \leq L \leq m$, the code $\mathcal{C}$ is $\left(\frac{L}{L+1} \left(1 - \frac{mR}{m-L+1}\right), L\right)$ list decodable.*

The following lemma explains why it is convenient to prove the stronger agreement notions rather than a proximity gap directly.

Lemma 4.13 ([[GG26](#)]). *Let $\mathcal{C} \subseteq \Sigma^N$ be an $\mathbb{F}$ linear code. Then the following statements hold.*

1. *If $\mathcal{C}$ has $(\ell, \delta, \varepsilon, \gamma)$ correlated agreement, then $\mathcal{C}$ has an $(\ell, \delta, \varepsilon, \gamma)$ proximity gap.*
2. *If $\mathcal{C}$ has $(\ell, \delta, \varepsilon)$ mutual correlated agreement, then $\mathcal{C}$ has $(\ell, \delta, \varepsilon, 0)$ perfect correlated agreement.*

This gives the following explicit finite parameter version of the correlated agreement and proximity gap bounds.

Theorem 4.14. *Let $\mathcal{C} \subseteq (\mathbb{F}_q^m)^N$ be either an m FRS code or an order m UM code of block length N and rate R . In the multiplicity-code case, assume that $q = p$ is prime and that the message degree bound $k = mRN$ satisfies $k < p$.*

Fix $\ell \geq 1$, $0 < \eta < 1$, and $1 \leq L, s \leq m$. Suppose that $q > \binom{L+1}{2}\ell$ and

$$0 < \rho < \min \left\{ \frac{L}{L+1} \left(1 - \frac{mR}{m-L+1} \right), 1 - \frac{mR}{m-s+1} - \frac{\ell + 1 - \frac{mR}{m-s+1}}{s+1} \right\}.$$

Define $\varepsilon_1 := \frac{L}{L+1} \left(1 - \frac{mR}{m-L+1} \right) - \rho > 0$, $\varepsilon_2 := 1 - \frac{mR}{m-s+1} - \rho > 0$, $t_1 := \ell + \left\lceil \frac{\ell \rho}{\varepsilon_1} \right\rceil$ and $A := \left\lfloor \frac{(t_1-1)(\rho(s-1)+\varepsilon_2)}{\varepsilon_2} \right\rfloor + 1$. Then:

1. *$\mathcal{C}$ has $\mathbb{F}_q$ - $\left(\ell, \rho, \frac{(\lceil \ell/\eta \rceil - 1)L + A}{q}, \eta\right)$ curve correlated agreement.*
2. *$\mathcal{C}$ has $\mathbb{F}_q$ - $\left(\ell, \rho, \frac{\lfloor \ell \rho N + \ell \rfloor L + A}{q}\right)$ mutual correlated agreement.*
3. *In particular, when $\ell = 1$, $\mathcal{C}$ has $\mathbb{F}_q$ - $\left(1, \rho, \frac{\lfloor \rho N + 1 \rfloor L + A}{q}, 0\right)$ perfect line correlated agreement.*

Proof. By [Theorem 4.12](#), for every $1 \leq L \leq m$, the code $\mathcal{C}$ is $\left(\frac{L}{L+1} \left(1 - \frac{mR}{m-L+1}\right), L\right)$ list decodable. By the definition of ε_1 , $\frac{L}{L+1} \left(1 - \frac{mR}{m-L+1}\right) = \rho + \varepsilon_1$. Moreover, since $t_1 - \ell = \left\lceil \frac{\ell\rho}{\varepsilon_1} \right\rceil$, we have $\rho \left(1 + \frac{\ell}{t_1 - \ell}\right) \leq \rho + \varepsilon_1$. Thus, the list decoding hypothesis needed in [Theorem 4.6](#) is satisfied with radius ρ and list size L .

Next, by [Theorem 4.11](#), $\mathcal{C}$ is a τ subspace design code with $\tau(s) = \frac{mR}{m-s+1}$. The assumption $\rho < 1 - \tau(s) - \frac{\ell+1-\tau(s)}{s+1}$ is precisely the radius condition required in [Theorem 4.5](#). Moreover, $1 - \tau(s) - \rho = \varepsilon_2$. Therefore, for every positive integer a , the code $\mathcal{C}$ is $\mathbb{F}_{q^*} \left(\ell, \rho, a, \left\lceil \frac{\varepsilon_2 a}{\rho(s-1)+\varepsilon_2} \right\rceil\right)$ curve decodable.

Now take $a = A$. By the definition of A , we have $A > \frac{(t_1-1)(\rho(s-1)+\varepsilon_2)}{\varepsilon_2}$. Hence, $\left\lceil \frac{\varepsilon_2 A}{\rho(s-1)+\varepsilon_2} \right\rceil \geq t_1$. Thus, $\mathcal{C}$ is $\mathbb{F}_{q^*}(\ell, \rho, A, t_1)$ curve decodable. Let $t_2 := \left\lceil \frac{\ell}{\eta} \right\rceil$. Applying [Corollary 4.10](#) with $\delta = \rho$, list size L , $a = A$, and this choice of t_2 gives curve correlated agreement with soundness $\frac{(t_2-1)L+A}{q}$ and slack ℓ/t_2 . Since $\frac{\ell}{t_2} \leq \eta$, this implies the claimed curve correlated agreement statement with slack η . The mutual correlated agreement statement follows from the second part of [Corollary 4.10](#), giving soundness $\frac{\lfloor \ell\rho N + \ell \rfloor L + A}{q}$.

Finally, setting $\ell = 1$ and applying [Lemma 4.13\(2\)](#) gives perfect line correlated agreement. Applying [Lemma 4.13\(1\)](#) then gives the corresponding line proximity gap. $\square$

Remark 4.15. The radius condition in [Theorem 4.14](#) can be expressed directly as a lower bound on the order parameter m . Whenever $L(1-R) - (L+1)\rho > 0$ and $s(1-R-\rho) - (\ell+\rho) > 0$, it is sufficient that

$$m \geq \max \left\{ \frac{L(L - (L+1)\rho - R)}{L(1-R) - (L+1)\rho}, \frac{s(s(1-\rho) - (\ell+\rho+R))}{s(1-R-\rho) - (\ell+\rho)} \right\}.$$

In particular, for $0 < \varepsilon < 1-R$ and $\rho \leq 1-R-\varepsilon$, choose $L := \lceil 2(1-R)/\varepsilon \rceil - 1$ and $s := \lceil 2(\ell+1-R)/\varepsilon \rceil - 1$. It then suffices to take $m \geq \frac{4R(\ell+1-R)}{\varepsilon^2} + \frac{2(\ell+1-2R)}{\varepsilon}$ under which the radius condition of [Theorem 4.14](#) holds for every $\rho \leq 1-R-\varepsilon$.

4.1 Optimal Proximity Gaps for Foldable FRS

For the folding applications, we use the evaluation points specified below.

Lemma 4.16. *Let q be a prime such that $2^n \mid q-1$ for some positive integer n . Let m, k be an positive integer and $t \mid (q-1)$ be an odd integer such that $t > \max\{k, m\}$. Let $N = 2^n$ and $r \leq n$ be an positive integer. Let $\alpha, \zeta \in \mathbb{F}_q^*$ such that $\text{ord}(\alpha) = N$ and $\text{ord}(\zeta) = t$. Then, the following is a m FRS code:*

$$\text{FRS}_{q, \frac{N}{2^r}, k, m}(\alpha, \zeta^{2^r}) := \left\{ \left(\left[\begin{array}{c} f(\alpha^{(i-1)2^r}) \\ f(\alpha^{(i-1)2^r} \zeta^{2^r}) \\ \vdots \\ f(\alpha^{(i-1)2^r} \zeta^{(m-1)2^r}) \end{array} \right] \right)_{i \in [\frac{N}{2^r}]} \mid f \in \mathbb{F}_q[x], \deg f < k \right\}.$$

Proof. The set of evaluation points used by the code above is

$$\mathcal{L}_r := \left\{ \alpha^{2^r i} \zeta^{2^r j} : 0 \leq i < \frac{N}{2^r}, 0 \leq j < m \right\}.$$

We show that $\mathcal{L}_r$ satisfies the requirements of an m FRS evaluation set. Because t is odd and $N = 2^n$, we have $\text{ord}(\zeta^{2^r}) = t$ and $\text{ord}(\alpha^{2^r}) = \frac{N}{2^r}$. In particular, the folding element $w = \zeta^{2^r}$ has order $t > k$, as required in [Definition 2.1](#).

It remains to show that all evaluation points are distinct. Assume not, and suppose that $\alpha^{2^r i} \zeta^{2^r j} = \alpha^{2^r i'} \zeta^{2^r j'}$ for some pair $(i, j) \neq (i', j')$. This implies that $\alpha^{2^r(i-i')} \zeta^{2^r(j-j')} = 1$. Since $\text{ord}(\alpha^{2^r}) = \frac{N}{2^r}$, $\text{ord}(\zeta^{2^r}) = t$ and $\gcd(N/2^r, t) = 1$, we have $\langle \alpha^{2^r} \rangle \cap \langle \zeta^{2^r} \rangle = \{1\}$. Therefore, $\alpha^{2^r(i-i')} = 1$ and $\zeta^{2^r(j-j')} = 1$, which imply $\frac{N}{2^r} \mid (i - i')$ and $t \mid (j - j')$. The conditions $0 \leq i, i' < \frac{N}{2^r}$ and $0 \leq j, j' < m < t$ force $i = i'$ and $j = j'$, a contradiction.

Hence all elements of $\mathcal{L}_r$ are distinct and the code above satisfies the m FRS code requirements in [Definition 2.1](#). $\square$

Lemma 4.17. *Under the assumptions of [Lemma 4.16](#), then $\text{FRS}_{q, \frac{N}{2^r}, k, m}(\alpha, \zeta^{2^r})$ is a τ_r subspace design code with $\tau_r(s) = \frac{2^r k}{(m-s+1)N}$ for every $1 \leq s \leq m$ and $\tau_r(s) = 1$ otherwise.*

Proof. Fix $1 \leq s \leq m$, and let $A \subseteq \text{FRS}_{q, \frac{N}{2^r}, k, m}(\alpha, \zeta^{2^r})$ be an $\mathbb{F}_q$ linear subspace of dimension $d \leq s$. The case $d = 0$ is immediate, so assume $d \geq 1$. Choose $f_1, \dots, f_d \in \mathbb{F}_q[X]_{<k}$ whose encodings form a basis of A . By Gaussian elimination on their coefficients and rescaling, we may choose these polynomials to be monic with distinct degrees $0 \leq e_1 < \dots < e_d < k$.

The coefficient of $X^{e_1 + \dots + e_d}$ in $W(X)$ is $\det((\zeta^{2^r e_j})^{i-1})_{i,j=1}^d = \prod_{1 \leq i < j \leq d} (\zeta^{2^r e_j} - \zeta^{2^r e_i})$. By [Lemma 4.16](#), $\text{ord}(\zeta^{2^r}) = t > k$, so the elements $\zeta^{2^r e_1}, \dots, \zeta^{2^r e_d}$ are pairwise distinct. Hence $W(X) \neq 0$ and $\deg W \leq d(k-1)$.

For each fixed i , we may choose the first $\dim_{\mathbb{F}_q}(A_i)$ basis polynomials so that their encodings form a basis of A_i . By the definition of A_i , the i -th folded symbol of each of these codewords is the zero vector. Hence, for every $h \leq \dim_{\mathbb{F}_q}(A_i)$, we have $f_h(\alpha^{(i-1)2^r} \zeta^{a2^r}) = 0$ for all $0 \leq a < m$.

Now fix $0 \leq j \leq m-d$ and evaluate the folded Wronskian matrix at $X = \alpha^{(i-1)2^r} \zeta^{j2^r}$. Its entry in row $a+1$ and column h is $f_h(\zeta^{a2^r} \alpha^{(i-1)2^r} \zeta^{j2^r}) = f_h(\alpha^{(i-1)2^r} \zeta^{(j+a)2^r})$ for $0 \leq a < d$. Since $j+a \leq m-1$, the preceding identity shows that this value is zero for every $h \leq \dim_{\mathbb{F}_q}(A_i)$. Therefore, every entry in the first $\dim_{\mathbb{F}_q}(A_i)$ columns, viewed as a polynomial in X , is divisible by $X - \alpha^{(i-1)2^r} \zeta^{j2^r}$. Consequently, $(X - \alpha^{(i-1)2^r} \zeta^{j2^r})^{\dim_{\mathbb{F}_q} A_i} \mid W(X)$.

All these points are distinct by [Lemma 4.16](#). Counting roots with multiplicity therefore gives $(m-d+1) \sum_{i=1}^{N/2^r} \dim_{\mathbb{F}_q}(A_i) \leq \deg W \leq d(k-1)$. Since $d \leq s$, we conclude that

$$\sum_{i=1}^{N/2^r} \dim_{\mathbb{F}_q}(A_i) \leq \frac{d(k-1)}{m-d+1} \leq \frac{dk}{m-s+1} = \frac{N}{2^r} \tau_r(s) \dim_{\mathbb{F}_q} A.$$

For $s > m$, the same inequality follows immediately from $\tau_r(s) = 1$ and $\dim_{\mathbb{F}_q}(A_i) \leq \dim_{\mathbb{F}_q} A$. This proves the claimed subspace design property. $\square$

Corollary 4.18. *For every $0 \leq r \leq n$, consider the m FRS code $\text{FRS}_{q, \frac{N}{2^r}, k, m}(\alpha, \zeta^{2^r})$ from Lemma 4.16. Suppose that the parameters satisfy the assumptions of Theorem 4.14 with block length $N/2^r$ and rate $\frac{2^r k}{mN}$. Then this code satisfies all the conclusions of Theorem 4.14 with these parameters.*

Proof. By Lemma 4.16, $\text{FRS}_{q, \frac{N}{2^r}, k, m}(\alpha, \zeta^{2^r})$ is an m FRS code of block length $N/2^r$, message length k and rate $\frac{2^r k}{mN}$.

Moreover, by Lemma 4.17, this code is a τ_r subspace design code with $\tau_r(s) = \frac{2^r k}{(m-s+1)N}$. Thus, its subspace design parameter has exactly the form required in Theorem 4.14. Together with the corresponding list decoding bound for FRS codes, all the assumptions of Theorem 4.14 are satisfied. Applying that theorem with block length $N/2^r$ and rate $2^r k/(mN)$ gives the claimed statements. $\square$

For 100-bit security, the minimum field size, proximity gap ε , and resulting soundness bounds for FRI [BCH⁺26], the previous FRS/UM analysis [GG26], and our refined analysis are presented in Table 6, Table 7, and Table 8, respectively. For all given parameters, the FRI soundness requires a field size exceeding 2^{128} . Compared with the previous FRS/UM analysis, our refinement reduces the required folding parameter m , field size, and proximity gap ε , while also yielding a tighter soundness bound.

Table 6: Smallest q for 100-bit Security from [BCH⁺26]

R	k	q -bit	ε	soundness	R	k	q -bit	ε	soundness
1/2	2^{16}	137	0.0272	100.31	1/8	2^{16}	137	0.0295	100.58
	2^{18}	139	0.0272	100.31		2^{18}	139	0.0295	100.58
	2^{20}	141	0.0272	100.31		2^{20}	141	0.0295	100.58
	2^{22}	143	0.0272	100.31		2^{22}	143	0.0295	100.58
	2^{24}	145	0.0272	100.31		2^{24}	145	0.0295	100.58
	2^{26}	147	0.0272	100.31		2^{26}	147	0.0295	100.58
	2^{28}	149	0.0272	100.31		2^{28}	149	0.0295	100.58
	2^{30}	151	0.0272	100.31		2^{30}	151	0.0295	100.58
1/4	2^{16}	137	0.0278	100.35	1/16	2^{16}	138	0.0250	100.29
	2^{18}	139	0.0278	100.35		2^{18}	140	0.0250	100.29
	2^{20}	141	0.0278	100.35		2^{20}	142	0.0250	100.29
	2^{22}	143	0.0278	100.35		2^{22}	144	0.0250	100.29
	2^{24}	145	0.0278	100.35		2^{24}	146	0.0250	100.29
	2^{26}	147	0.0278	100.35		2^{26}	148	0.0250	100.29
	2^{28}	149	0.0278	100.35		2^{28}	150	0.0250	100.29
	2^{30}	151	0.0278	100.35		2^{30}	152	0.0250	100.29

Table 7: Smallest q for 100-bit Security from [GG26]

R	k	m	q -bit	ε	soundness	R	k	m	q -bit	ε	soundness
1/2	2 ¹⁶	128	113	0.4000	100.57	1/8	2 ¹⁶	64	143	0.6667	128.41
	2 ¹⁸	128	115	0.4000	100.65		2 ¹⁸	64	145	0.6667	128.41
	2 ²⁰	128	117	0.4000	100.67		2 ²⁰	64	147	0.6667	128.41
	2 ²²	128	119	0.4000	100.68		2 ²²	64	149	0.6667	128.41
	2 ²⁴	128	121	0.4000	100.68		2 ²⁴	64	151	0.6667	128.42
	2 ²⁶	128	123	0.4000	100.68		2 ²⁶	64	153	0.6667	128.42
	2 ²⁸	128	125	0.4000	100.68		2 ²⁸	64	155	0.6667	128.42
	2 ³⁰	128	127	0.4000	100.68		2 ³⁰	64	157	0.6667	128.42
1/4	2 ¹⁶	128	141	0.6667	128.40	1/16	2 ¹⁶	64	144	0.6667	128.41
	2 ¹⁸	128	143	0.6667	128.41		2 ¹⁸	64	146	0.6667	128.41
	2 ²⁰	128	145	0.6667	128.41		2 ²⁰	64	148	0.6667	128.41
	2 ²²	128	147	0.6667	128.41		2 ²²	64	150	0.6667	128.41
	2 ²⁴	128	149	0.6667	128.41		2 ²⁴	64	152	0.6667	128.42
	2 ²⁶	128	151	0.6667	128.42		2 ²⁶	64	154	0.6667	128.42
	2 ²⁸	128	153	0.6667	128.42		2 ²⁸	64	156	0.6667	128.42
	2 ³⁰	128	155	0.6667	128.42		2 ³⁰	64	158	0.6667	128.42

Table 8: Smallest q for 100-bit Security from Our Results

smallest q first and then smallest m and then the total proof size											
R	k	m	q -bit	ε	soundness	R	k	m	q -bit	ε	soundness
1/2	2 ¹⁶	32	110	0.2890	100.16	1/8	2 ¹⁶	16	115	0.3760	100.00
	2 ¹⁸	32	112	0.2890	100.22		2 ¹⁸	16	117	0.3760	100.00
	2 ²⁰	32	114	0.2890	100.24		2 ²⁰	16	119	0.3760	100.00
	2 ²²	32	116	0.2890	100.24		2 ²²	16	121	0.3760	100.00
	2 ²⁴	32	118	0.2890	100.24		2 ²⁴	16	123	0.3760	100.00
	2 ²⁶	32	120	0.2890	100.24		2 ²⁶	16	125	0.3760	100.00
	2 ²⁸	32	122	0.2890	100.24		2 ²⁸	16	127	0.3760	100.00
	2 ³⁰	32	124	0.2890	100.24		2 ³⁰	16	129	0.3760	100.00
1/4	2 ¹⁶	16	113	0.3770	100.18	1/16	2 ¹⁶	8	118	0.3670	100.81
	2 ¹⁸	16	115	0.3760	100.29		2 ¹⁸	8	120	0.3670	100.81
	2 ²⁰	16	117	0.3760	100.39		2 ²⁰	8	122	0.3670	100.81
	2 ²²	16	119	0.3760	100.41		2 ²²	8	124	0.3670	100.81
	2 ²⁴	16	121	0.3760	100.42		2 ²⁴	8	126	0.3670	100.81
	2 ²⁶	16	123	0.3760	100.42		2 ²⁶	8	128	0.3670	100.81
	2 ²⁸	16	125	0.3760	100.42		2 ²⁸	8	130	0.3670	100.81
	2 ³⁰	16	127	0.3760	100.42		2 ³⁰	8	132	0.3670	100.81

References

- ACFY24. Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. STIR: reed-solomon proximity testing with fewer queries. In *CRYPTO 2024*, volume 14929 of *LNCS*, pages 380–413. Springer, 2024. 2

- ACFY25. Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: reed-solomon proximity testing with super-fast verification. In *EUROCRYPT 2025*, volume 15604 of *LNCS*, pages 214–243. Springer, 2025. [2](#)
- AHIV17. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkita-subramaniam. Liger: Lightweight sublinear arguments without a trusted setup. In *CCS 2017*, pages 2087–2104. ACM, 2017. [2](#)
- BBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018*, volume 107 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. [2](#), [5](#)
- BCF⁺25. Martijn Brehm, Binyi Chen, Ben Fisch, Nicolas Resch, Ron D. Rothblum, and Hadas Zeilberger. Blaze: Fast snarks from interleaved RAA codes. In *EUROCRYPT 2025*, volume 15604 of *LNCS*, pages 123–152. Springer, 2025. [2](#), [3](#)
- BCG⁺17. Jonathan Bootle, Andrea Cerulli, Essam Ghadafi, Jens Groth, Mohammad Hajiabadi, and Sune K. Jakobsen. Linear-time zero-knowledge proofs for arithmetic circuit satisfiability. In *ASIACRYPT 2017*, volume 10626 of *LNCS*, pages 336–365. Springer, 2017. [2](#)
- BCH⁺26. Eli Ben-Sasson, Dan Carmon, Ulrich Haböck, Swastik Kopparty, and Shubhangi Saraf. On proximity gaps of reed-solomon codes. In Aditya Bhaskara and Artur Czumaj, editors, *Proceedings of the 58th Annual ACM Symposium on Theory of Computing, STOC 2026, Salt Lake City, UT, USA, June 22–26, 2026*, pages 1157–1167. ACM, 2026. [2](#), [3](#), [26](#), [37](#)
- 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. [2](#)
- BCL22. Jonathan Bootle, Alessandro Chiesa, and Siqi Liu. Zero-knowledge iops with linear-time prover and polylogarithmic-time verifier. In *EUROCRYPT 2022*, volume 13276 of *LNCS*, pages 275–304. Springer, 2022. [2](#)
- BCS16. Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam D. Smith, editors, *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 31–60, 2016. [14](#), [26](#)
- BFK⁺24. Alexander R. Block, Zhiyong Fang, Jonathan Katz, Justin Thaler, Hendrik Waldner, and Yupeng Zhang. Field-agnostic snarks from expand-accumulate codes. In *CRYPTO 2024*, volume 14929 of *LNCS*, pages 276–307. Springer, 2024. [2](#), [3](#)
- BGK⁺23. Alexander R. Block, Albert Garreta, Jonathan Katz, Justin Thaler, Pratyush Ranjan Tiwari, and Michał Zając. Fiat-shamir security of FRI and related SNARKs. In *ASIACRYPT 2023*, pages 3–40. Springer, 2023. [21](#)
- BGKS20. Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. DEEP-FRI: Sampling outside the box improves soundness. In *ITCS*, volume 151 of *LIPIcs*, pages 5:1–5:32. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. [2](#), [3](#)
- BHKS21. Siddharth Bhandari, Prahladh Harsha, Mrinal Kumar, and Madhu Sudan. Ideal-theoretic explanation of capacity-achieving decoding. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021*,

- University of Washington, Seattle, Washington, USA (Virtual Conference), August 16-18, 2021, volume 207 of *LIPICs*, pages 56:1–56:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. [4](#), [10](#), [14](#)
- 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, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1082–1090. ACM, 2019. [14](#)
- CZ25. Yeyuan Chen and Zihan Zhang. Explicit folded reed-solomon and multiplicity codes achieve relaxed generalized singleton bounds. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1–12, 2025. [12](#), [33](#), [34](#), [41](#)
- GG26. Rohan Goyal and Venkatesan Guruswami. Optimal proximity gaps for subspace-design codes and (random) reed-solomon codes. In *Proceedings of the 58th Annual ACM Symposium on Theory of Computing*, pages 1558–1568, 2026. [2](#), [3](#), [4](#), [5](#), [10](#), [11](#), [12](#), [13](#), [30](#), [32](#), [34](#), [37](#), [38](#), [41](#)
- GHKS24. Rohan Goyal, Prahladh Harsha, Mrinal Kumar, and Ashutosh Shankar. Fast list decoding of univariate multiplicity and folded reed-solomon codes. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*, pages 328–343. IEEE, 2024. [41](#)
- GK16. Venkatesan Guruswami and Swastik Kopparty. Explicit subspace designs. *Combinatorica*, 36(2):161–185, 2016. [12](#), [33](#), [34](#)
- GLS⁺23. Alexander Golovnev, Jonathan Lee, Srinath T. V. Setty, Justin Thaler, and Riad S. Wahby. Brakedown: Linear-time and field-agnostic snarks for R1CS. In *CRYPTO 2023*, volume 14082 of *LNCS*, pages 193–226. Springer, 2023. [2](#), [3](#)
- GMW25. Albert Garreta, Nicolas Mohnblatt, and Benedikt Wagner. A simplified round-by-round soundness proof of FRI. *IACR Cryptol. ePrint Arch.*, 2025:1993, 2025. [21](#), [22](#)
- GNR26. Kobi Gurkan, Andrija Novakovic, and Ron D. Rothblum. Bolt: Faster snarks from sketched codes. In Nadia Heninger and Mike Rosulek, editors, *Advances in Cryptology - CRYPTO 2026*, volume 16808 of *LNCS*, pages 396–427. Springer, 2026. [3](#)
- GR08. Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on information theory*, 54(1):135–150, 2008. [11](#)
- GSS00. Venkatesan Guruswami, Amit Sahai, and Madhu Sudan. "soft-decision" decoding of chinese remainder codes. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, Redondo Beach, California, USA, November 12-14, 2000*, pages 159–168. IEEE Computer Society, 2000. [4](#), [14](#)
- Jer26. Fernando Granha Jeronimo. Algorithmic list decoding at capacity and optimal proximity gaps for reed-solomon codes, 2026. [5](#)
- Kop15. Swastik Kopparty. List-decoding multiplicity codes. *Theory of Computing*, 11(1):149–182, 2015. [12](#), [41](#)
- KPR⁺26. Vladimir Kolesnikov, Stanislav Peceny, Rahul Rachuri, Srinivasan Raghuraman, Peter Rindal, and Harshal Shah. Block-accumulate codes: Accelerated linear codes for pcgs and ZK. In Nadia Heninger and Mike Rosulek, editors, *Advances in Cryptology - CRYPTO 2026*, volume 16807 of *LNCS*, pages 397–429. Springer, 2026. [3](#)

- KRSW18. Swastik Kopparty, Noga Ron-Zewi, Shubhangi Saraf, and Mary Wootters. Improved decoding of folded reed-solomon and multiplicity codes. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 212–223. IEEE Computer Society, 2018. [41](#)
- RR22. Noga Ron-Zewi and Ron D. Rothblum. Proving as fast as computing: succinct arguments with constant prover overhead. In *STOC 2022*, pages 1353–1363. ACM, 2022. [2](#), [3](#)
- RT97. M Yu Rosenbloom and Michael Anatol’evich Tsfasman. Codes for the m-metric. *Problemy Peredachi Informatsii*, 33(1):55–63, 1997. [12](#)
- Spi96. Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Trans. Inf. Theory*, 42(6):1723–1731, 1996. [3](#)
- vzGG13. Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra (3. ed.)*. Cambridge University Press, 2013. [15](#), [24](#)
- XZS22. Tiancheng Xie, Yupeng Zhang, and Dawn Song. Orion: Zero knowledge proof with linear prover time. In *CRYPTO 2022*, volume 13510 of *LNCS*, pages 299–328. Springer, 2022. [2](#)
- ZCF24. Hadas Zeilberger, Binyi Chen, and Ben Fisch. Basefold: Efficient field-agnostic polynomial commitment schemes from foldable codes. In *CRYPTO 2024*, volume 14929 of *LNCS*, pages 138–169. Springer, 2024. [2](#)

Supplementary Material

A Folding Univariate Multiplicity Codes from Derivatives

In coding theory papers concerning list-decoding [[Kop15](#), [KRSW18](#), [GHKS24](#), [CZ25](#)] and proximity gaps [[GG26](#)], univariate multiplicity codewords are represented via higher order derivatives rather than the polynomial forms. The problem is formulated as follows:

Problem A.1. Let $f(x) \in \mathbb{F}^{<k}[x]$ and suppose that $f(x) = f_0(x^2) + xf_1(x^2)$ is its unique even/odd decomposition of f . Given

$$\left\{ f^{(j)}(\pm a) \right\}_{j \in [0, m-1]}$$

for some $a \in \mathbb{F}^*$ and a challenge γ , the goal is to compute

$$\left\{ f_0^{(i)}(a^2) + \gamma \cdot f_1^{(i)}(a^2) \right\}_{i \in [0, m-1]}.$$

We show that this task can be performed in $O(M(m))$ field operations, where $M(m)$ denotes the cost of multiplying two degree m polynomials over $\mathbb{F}$. Throughout this subsection, we assume $\text{char}(\mathbb{F}) \neq 2$ and $a \neq 0$.

The key idea is to work in the truncated formal power series ring

$$\mathbb{F}[[u]] / (u^m).$$

The derivative evaluations at $\pm a$ naturally determine the two Taylor expansions

$$f(u+a) \equiv \sum_{k=0}^{m-1} \frac{f^{(k)}(a)}{k!} u^k \pmod{u^m} \text{ and } f(u-a) \equiv \sum_{k=0}^{m-1} \frac{f^{(k)}(-a)}{k!} u^k \pmod{u^m}.$$

Equivalently, after the change of variables $u = x - a$ and $u = x + a$, respectively,

$$f(x) \equiv \sum_{k=0}^{m-1} \frac{f^{(k)}(a)}{k!} (x-a)^k \pmod{(x-a)^m} \text{ and } f(x) \equiv \sum_{k=0}^{m-1} \frac{f^{(k)}(-a)}{k!} (x+a)^k \pmod{(x+a)^m}.$$

Thus, from the given derivative evaluations $\{f^{(j)}(\pm a)\}_{j \in [0, m-1]}$, we can construct the two local representatives

$$G(x) := f(x) \pmod{(x-a)^m} \text{ and } H(x) := f(x) \pmod{(x+a)^m}$$

with $\deg(G), \deg(H) < m$. Converting the coefficients of G and H from the shifted bases

$$\{(x-a)^j\}_{j \in [0, m-1]} \text{ and } \{(x+a)^j\}_{j \in [0, m-1]}$$

to the standard monomial basis x^j $_{j \in [0, m-1]}$ is a Taylor shift operation and takes $O(M(m))$ field operations.

Our goal is to recover the truncated Taylor expansions

$$f_0(u+a^2) \equiv \sum_{k=0}^{m-1} \frac{f_0^{(k)}(a^2)}{k!} u^k \pmod{u^m} \text{ and } f_1(u+a^2) \equiv \sum_{k=0}^{m-1} \frac{f_1^{(k)}(a^2)}{k!} u^k \pmod{u^m}.$$

The desired derivatives

$$\{f_0^{(i)}(a^2), f_1^{(i)}(a^2)\}_{i \in [0, m-1]}$$

can then be read off from their coefficients.

To this end, let

$$r(u) := \sqrt{u+a^2} \in \mathbb{F}[[u]],$$

which satisfies

$$r(u)^2 = u + a^2 \text{ and } r(0) = a.$$

Then, $r(u)$ is invertible in $\mathbb{F}[[u]]$ as $a \neq 0$. Moreover, the truncated series $r(u)$ and its inverse can be computed in $O(M(m))$ field operations according to [Lemma 3.12](#).

By the decomposition $f(x) = f_0(x^2) + x f_1(x^2)$, we have

$$\begin{cases} f(r(u)) = f_0(u+a^2) + r(u) \cdot f_1(u+a^2) \\ f(-r(u)) = f_0(u+a^2) - r(u) \cdot f_1(u+a^2) \end{cases}.$$

Consequently,

$$\begin{cases} f_0(u+a^2) = \frac{f(r(u)) + f(-r(u))}{2} \pmod{u^m} \\ f_1(u+a^2) = \frac{f(r(u)) - f(-r(u))}{2r(u)} \pmod{u^m} \end{cases}, \quad (9)$$

Therefore, it suffices to compute

$$f(r(u)) \pmod{u^m} \text{ and } f(-r(u)) \pmod{u^m}.$$

The following lemma allows us to replace f by its local representatives G and H .

Lemma A.2. *Suppose that*

$$G(x) \equiv f(x) \pmod{(x-a)^m} \text{ and } H(x) \equiv f(x) \pmod{(x+a)^m}.$$

Then, in $\mathbb{F}[[u]]/(u^m)$,

$$G(r(u)) \equiv f(r(u)) \pmod{u^m} \text{ and } H(-r(u)) \equiv f(-r(u)) \pmod{u^m}.$$

Proof. Assume that there exist $R(x), S(x) \in \mathbb{F}[x]$ such that

$$G(x) - f(x) = (x-a)^m \cdot R(x) \text{ and } H(x) - f(x) = (x+a)^m \cdot S(x).$$

Substituting $x = \pm r(u)$ gives

$$G(r(u)) - f(r(u)) = (r(u) - a)^m \cdot R(r(u)) \quad (10)$$

and

$$\begin{aligned} H(-r(u)) - f(-r(u)) &= (-r(u) + a)^m \cdot S(-r(u)) \\ &= (-1)^m (r(u) - a)^m \cdot S(-r(u)), \end{aligned} \quad (11)$$

respectively. It remains to observe that

$$r(u) - a = \frac{u}{r(u) + a}.$$

since $r(0) + a = 2a \neq 0$, the series $r(u) + a$ is invertible in $\mathbb{F}[[u]]$. Hence,

$$r(u) - a \in u\mathbb{F}[[u]] \text{ and } (r(u) - a)^m = O(u^m) \in u^m\mathbb{F}[[u]].$$

As a matter of fact,

$$r(u) - a = \sqrt{u + a^2} - a = \frac{u}{2a} - \frac{u^2}{8a^3} + \cdots \in u\mathbb{F}[[u]].$$

Thus, both differences in [Eq.\(10\)](#) and [Eq.\(11\)](#) vanish modulo u^m , proving

$$G(r(u)) \equiv f(r(u)) \pmod{u^m} \text{ and } H(-r(u)) \equiv f(-r(u)) \pmod{u^m}.$$

□

It remains to evaluate $G(r(u))$ and $H(-r(u))$. Since $\deg(G), \deg(H) < m$, each admits a unique even/odd decomposition

$$G(x) = G_0(x^2) + xG_1(x^2) \text{ and } H(x) = H_0(x^2) + xH_1(x^2),$$

where

$$\deg(G_0), \deg(G_1), \deg(H_0), \deg(H_1) < m/2.$$

Substituting $x = r(u)$ and $x = -r(u)$, respectively, yields

$$\begin{cases} G(r(u)) = G_0(u + a^2) + r(u) \cdot G_1(u + a^2) \mod u^m \\ H(-r(u)) = H_0(u + a^2) - r(u) \cdot H_1(u + a^2) \mod u^m. \end{cases}$$

The four shifted polynomials

$$G_0(u + a^2), G_1(u + a^2), H_0(u + a^2), H_1(u + a^2)$$

can each be computed by a Taylor shift. Since their degrees are at most $m/2$, the total cost remains $O(M(m))$. Multiplying $G_1(u + a^2)$ and $H_1(u + a^2)$ by $r(u)$ modulo u^m then gives

$$G(r(u)) \mod u^m \text{ and } H(-r(u)) \mod u^m.$$

By [Lemma A.2](#), there are precisely

$$f(r(u)) \mod u^m \text{ and } f(-r(u)) \mod u^m.$$

Equation [Eq.\(9\)](#) therefore recovers

$$(f_0(u + a^2), f_1(u + a^2)).$$

Finally, extracting the coefficients of u^k and multiplying by $k!$ gives the desired derivatives.

Procedure 9: Folding UMCs from Derivatives

1. Construct $G(x), H(x) \in \mathbb{F}^{<m}[x]$ from the derivatives such that $//O(M(m))$

$$G(x) \equiv f(x) \mod (x - a)^m \text{ and } H(x) \equiv f(x) \mod (x + a)^m.$$

2. Let

$$r(u) := \sqrt{u + a^2}.$$

be the formal power series with constant term a . Compute $//O(M(m))$

$$r(u) \mod u^m \text{ and } \frac{1}{r(u)} \mod u^m.$$

3. Compute the even/odd decompositions $//O(m)$

$$G(x) = G_0(x^2) + xG_1(x^2) \text{ and } H(x) = H_0(x^2) + xH_1(x^2).$$

Perform Taylor shifts to obtain $//O(M(m))$

$$G_0(u + a^2), G_1(u + a^2), H_0(u + a^2), H_1(u + a^2).$$

4. Compute $//O(M(m))$

$$G(r(u)) := G_0(u + a^2) + r(u)G_1(u + a^2)$$

and

$$H(-r(u)) := H_0(u + a^2) - r(u)G_1(u + a^2).$$

By [Lemma A.2](#),

$$G(r(u)) \equiv f(r(u)) \pmod{u^m} \text{ and } H(r(u)) \equiv f(-r(u)) \pmod{u^m}.$$

5. Recover the folded polynomials according to $//O(M(m))$

$$\begin{cases} f_0(u + a^2) = \frac{f(r(u)) + f(-r(u))}{2} \pmod{u^m} \\ f_1(u + a^2) = \frac{f(r(u)) - f(-r(u))}{2r(u)} \pmod{u^m} \end{cases}$$

6. Read off the derivatives and form the desired linear combinations: $//O(m)$

$$\left\{ k! \cdot ([u^k]f_0(u + a^2) + \gamma \cdot [u^k]f_1(u + a^2)) \right\}_{k \in [0, m-1]}.$$

Thus, all steps require a total of $O(M(m))$ field operations. In particular, the procedure converts the derivative representation

$$\left\{ f^{(j)}(\pm a) \right\}_{j \in [0, m-1]}$$

into the folded derivative representation

$$\left\{ f_0^{(i)}(a^2) + \gamma \cdot f_1^{(i)}(a^2) \right\}_{i \in [0, m-1]}.$$

without explicitly reconstructing the underlying polynomial f .