Elliptic Curve Fast Fourier Transform (ECFFT) Part II: Scalable and Transparent Proofs over All Large Fields

Eli Ben-Sasson* Dan Carmon* Swastik Kopparty† David Levit*

August 30, 2022

Abstract

Concretely efficient interactive oracle proofs (IOPs) are of interest due to their applications to scaling blockchains, their minimal security assumptions, and their potential future-proof resistance to quantum attacks.

Scalable IOPs, in which prover time scales quasilinearly with the computation size and verifier time scales poly-logarithmically with it, have been known to exist thus far only over a set of finite fields of negligible density, namely, over "FFT-friendly" fields that contain a sub-group of size 2^k.

Our main result is to show that scalable IOPs can be constructed over *any* sufficiently large finite field, of size that is at least quadratic in the length of computation whose integrity is proved by the IOP. This result has practical applications as well, because it reduces the proving and verification complexity of cryptographic statements that are naturally stated over pre-defined finite fields which are not "FFT-friendly".

Prior state-of-the-art scalable IOPs relied heavily on arithmetization via univariate polynomials and Reed–Solomon codes over FFT-friendly fields. To prove our main result and extend scalability to all large finite fields, we generalize the prior techniques and use new algebraic geometry codes evaluated on sub-groups of elliptic curves (elliptic curve codes). We also show a new arithmetization scheme that uses the rich and well-understood group structure of elliptic curves to reduce statements of computational integrity to other statements about the proximity of functions evaluated on the elliptic curve to the new family of elliptic curve codes.

This paper continues our recent work [BCKL21] that used elliptic curves and their subgroups to create FFT-based algorithms for polynomial manipulation over generic finite fields. However, our new IOP constructions force us to use new codes (ones that are not based on polynomials), and this poses a new set of challenges involving the more restricted automorphism group of these codes, and the constraints of Riemann–Roch spaces of strictly positive genus.

*StarkWare Industries Ltd. {eli,dancar,david}@starkware.co

†Department of Mathematics and Department of Computer Science, University of Toronto. swastik.kopparty@gmail.com

1 Introduction

Arithmetization was first used to construct interactive proofs in the seminal work of Lund et al. [LFKN92] and shortly after played a key role in Shamir's proof of $\mathsf{IP} = \mathsf{PSPACE}$ [Sha92]. Ever since, this invaluable tool has dominated the construction of interactive proofs (IP), multiprover interactive proofs (MIP), zero knowledge proofs (ZK), probabilistically checkable proofs (PCP) and related protocols. Arithmetization reduces statements about computational integrity, like

> *"I processed* $\mathsf{T} = 10,000$ *valid Ethereum transactions, leading to new Ethereum state* S*"*

to completely different statements, about low degree polynomials over a finite field $\mathbb{F}$, like

> *"I know polynomials* $A(X), B(X)$ *over finite field* $\mathbb{F}$ *of degree at most* T *that satisfy a set of polynomial constraints".*

The question studied in this paper is: Which finite fields $\mathbb{F}$ can be used to create transparent[1], scalable and concretely efficient proof systems? We start by surveying the existing state of the art in this area.

To reach polynomial efficiency, any large finite field suffices. Early uses of arithmetization, for example, in the seminal proofs of (i) $\mathsf{MIP} = \mathsf{NEXP}$ [BFL91], (ii) the poly-logarithmic verification of NP [BFLS91] and (iii) the PCP Theorem [ALM+98, AS98], all work with *any* sufficiently large finite field, of size at least $\mathsf{poly}(\mathsf{T})$, where T denotes the length of the (nondeterministic) computation whose integrity is being proved; in the case of the PCP Theorem, a field of size $\mathrm{polylog}(\mathsf{T})$ suffices. The communication complexity in all of these celebrated protocols is extremely efficient — at most poly-logarithmic in T. However, none of these early constructions were ever deployed in practice because their proofs, although of polynomial length in T, were of impractical size, and the arithmetic complexity of both prover and verifier were, concretely, prohibitively large.

Scalable proof systems over FFT-friendly fields. The situation changed dramatically, in terms of both efficiency and field type, with the advent of *scalable* information theoretic proof systems. A proof system is called *scalable* when both (i) proving time[2] scales quasilinearly in T and, simultaneously (ii) verification time scales poly-logarithmically in T (and polynomially in the description of the computation whose integrity is proved); see [BBHR19, Definition 3.3] for an exact definition. Scalable PCP systems for any language in NEXP were presented by [BS08, BGH+05, BCGT13], improving proving time from $\mathsf{T}^{O(1)}$ to $\mathsf{T}\,\mathrm{polylog}\,\mathsf{T}$. However, these constructions limited $\mathbb{F}$ to be *FFT-friendly* which means it must contain a sub-group of size 2^{k}, for integer k (the group can be multiplicative or additive)[3]. In spite of their improved efficiency, scalable PCPs are not used in practice because the exponents in the poly-log expressions for proving and verification time, and the amortized soundness error per PCP-query, are still, practically speaking, too large.

The last and final step needed to create concretely efficient proof systems for NEXP was taken within a relatively new computational model, the interactive oracle proof (IOP) model [BCS16, RRR16] that generalizes both IP and PCP. From a computational complexity point of view, $\mathsf{IOP} = \mathsf{MIP} = \mathsf{PCP} = \mathsf{NEXP}$ (see [BCS16]). Within this model, proving time was reduced to $O(\mathsf{T}\log\mathsf{T})$ and verification time to $O(\log\mathsf{T})$, with relatively small asymptotic constants [BBHR19]. The requirement that $\mathbb{F}$ be FFT-friendly fields remained.

To summarize, early $\mathsf{IP}, \mathsf{MIP}$ and PCP constructions work over any sufficiently large finite field, but *scalable* PCPs and IOPs required FFT-friendly ones. This raises the question of whether FFT-friendliness is needed for scalability, and sets the ground for our main result.

[1] A proof system is transparent when all verifier messages are public random coins; such systems are also called Arthur Merlin protocols.

[2] Unless mentioned otherwise, throughout the paper running time is measured in number of field operations, i.e., we assign unit cost to arithmetic operations over the finite field.

[3] More generally, scalable PCPs and IOPs can be constructed over any $\mathbb{F}$ which has a sub-group of size that is a product of small primes, but prover and verifier running time increase as the prime factors increase in number and size. For simplicity we stick to interpreting an FFT-friendly field as one containing a multiplicative subgroup of size 2^{k}.

1.1 Main Results

The language most naturally suited for creating scalable IOPs is that of arithmetic intermediate representations (AIR) [BBHR19, Sta21]. Informally, an AIR instance of complexity m and length T is defined over a finite field $\mathbb{F}$ by a set of low-degree multivariate constraints, described by arithmetic circuits whose total sum (number of gates) is m, and by a cyclic group D of size T (see Definition 2.1). An AIR witness is a tuple of functions $f_1, \ldots, f_w : \mathsf{D} \to \mathbb{F}$ (see Definition 2.4), and the AIR instance is satisfied by it if the application of the polynomial constraints to the functions $f_1, \ldots, f_w$ and various cyclic shifts of them satisfy the polynomial constraints of the AIR instance (see Definition 2.5).

From a concrete complexity point of view, the language of AIRs is used to define computational integrity statements for scalable and transparent argument of knowledge (STARK) systems, directly for specific computations like hashing with ethSTARK [Sta21], for domain specific languages like Winterfell, and for universal (Turing complete) virtual machines like Cairo [GPR21]. In all these cases, the computations and virtual machines are specified by AIRs. Systems written over these machines, like StarkEx, have been used to process millions of transactions and billions of dollars on Ethereum, underscoring their practical relevance.

From an asymptotic complexity point of view, the language of satisfiable AIR instances is complete for NEXP. When restricting AIR to FFT-friendly fields, the ensuing sub-language (FFT-friendly-AIR) remains NEXP-complete. As mentioned earlier, prior to this work, it was known that the language of FFT-friendly-AIR has a *strictly* scalable and transparent IOP [BBHR19]. By strictly scalable we mean that (i) prover complexity is $\mathsf{T} \cdot (O(\log \mathsf{T}) + \mathsf{poly}(m))$ and, simultaneously, (ii) verifier complexity is $O(\log \mathsf{T}) + \mathsf{poly}(m)$, i.e., the exponents in all $\mathrm{polylog}$ expressions are 1.

The main result of this paper is to remove the FFT-friendly requirement about fields, leading to the following statement.

Theorem 1.1 (Main Theorem — Informal)**.** *For any finite field $\mathbb{F}$ and $\mathsf{T} \leq \sqrt{|\mathbb{F}|}$, the satisfiability of AIR instances over $\mathbb{F}$ of size m and computation length at most T can be verified by a strictly scalable and transparent IOP of knowledge with advice*[4] *In particular, there exist randomized procedures for proving and verification that require $\mathsf{T} \cdot (O(\log \mathsf{T}) + \mathsf{poly}(m))$ arithmetic operations over $\mathbb{F}$ for proving, and $\lambda \cdot (O(\log \mathsf{T}) + \mathsf{poly}(m))$ arithmetic operations over $\mathbb{F}$ for verification with knowledge soundness error at most $2^{-\lambda}$.*

We point out that our results apply to other NEXP complete languages for succinct IOPs, such as the succinct R1CS systems used in [BCG+19]; due to the concrete considerations mentioned above, as well as space limitations, we focus only on AIR.

Remark 1.2 (Zero Knowledge). The construction used in Theorem 1.1 can be augmented to achieve perfect zero knowledge, just like the FFT-friendly version of it (Theorem 2.8) can be augmented to an IOP with perfect zero knowledge [BCF+17]. We omit the addition of zero knowledge from this version due to space limitations.

Remark 1.3 (Post-quantum security). A number of works have shown that applying the Kilian-Micali and/or the BCS transformation from IOPs to noninteractive arguments are secure in the quantum random oracle model, and these generic transformations apply to all our results, rendering them post-quantum secure in this model [COS20, CMS19, CMSZ21].

Fast IOPs of Proximity for Reed–Solomon and Elliptic Curve codes A major step, and bottleneck, in the construction IOPs and PCPs is that of low-degree testing. This is the sub-protocol that is given oracle access to a function $f : \mathsf{D}' \to \mathbb{F}$ and is charged with distinguishing between the case that f is a low-degree polynomial, i.e., a Reed–Solomon (RS) codeword, and the case that f is far, in Hamming distance, from the RS code. Strictly scalable IOPs use the Fast RS IOPP (FRI) [BBHR18] protocol targeted for RS codes. For a function of blocklength $n = |\mathsf{D}'|$, the FRI protocol guarantees linear proving time ($O(n)$ arithmetic operations), strictly logarithmic verification time and query complexity ($O(\lambda \log n)$ arithmetic operations, to reduce the soundness error to $2^{-\lambda}$).

One of the main reasons that until now scalable IOPs were limited to FFT-friendly fields was the fact that the FRI protocol is tightly related to the FFT algorithm, and can be described as "randomly folding" an FFT. As part of our

[4] The proving and verifying procedures depend on $O(\mathsf{T} \log q)$ bits of advice that depend only on $|\mathbb{F}|$ and T – furthermore, this advice can be generated by a randomized algorithm in time $O(\mathsf{T}\,\mathrm{polylog}(\mathsf{T} \cdot q))$ with high probability.

proof of Theorem 1.1 we also extend the FRI protocol from [BBHR18], and its analysis from [BCI+20], to hold over all fields, provided $|\mathbb{F}| \geq \Omega(\sqrt{n})$.

Theorem 1.4 (FRI over all fields, informal)**.** *For any finite field $\mathbb{F}$ of size q, integer n a power of 2 satisfying $n \leq \sqrt{q}$, integer t and integer $\mathcal{R}$, the following holds.*

There exists a subset $\mathsf{D}' \subseteq \mathbb{F}, |\mathsf{D}'| = n$, such that the family of RS codes of rate[5] $\rho = 2^{-\mathcal{R}}$ evaluated over D' has an IOP of proximity with:

- *$O(n)$ proving complexity,*
- *$O(t \cdot \log n)$ verification complexity,*
- *$t \cdot \log n$ query complexity,*
- *the following soundness behavior: if f is δ-far in Hamming distance from the code, the probability that f is accepted by the protocol is at most $\left(\max\left\{(1-\delta), \sqrt{\rho}\right\} - o(1)\right)^t$.*

See Section 2.3 for more details and a formal statement of the result above.

We point out that we also obtain (and need, to prove Theorem 1.1) an IOPP for a more general family of codes – that comprised of evaluations of functions over certain carefully selected points on an elliptic curve E; the points of evaluation are cosets of a cyclic group of size 2^k inside the elliptic curve group. We call this protocol an *elliptic curve* FRI, abbreviated EC-FRI, because the IOPP for this family of elliptic curve codes works by "decomposing" a function on the elliptic curve into a pair of RS codewords and applying Theorem 1.4 to this pair. See Section 6.3 for details.

Applications to concrete scalability We briefly argue why Theorem 1.1 is interesting from the point of view of concrete (rather than asymptotic) complexity, in applied cryptography settings. There are quite a few cryptographic primitives used in practice that are naturally defined over specific, and non-FFT-friendly, finite fields. Examples include the NIST Curve P-256 (used, e.g., on Apple smartphones) and the secp256k1 curve (used for Bitcoin signatures), both of which are prime, non-FFT-friendly, fields. Consider a prover attempting to prove she processed correctly a large batch of ECDSA signatures over either one of these primes, denoting it by p. Today, the prover would need to arithmetize her statement over some FFT-friendly field, and thus simulate the basic arithmetic operations of the (non-FFT friendly) field $\mathbb{F}_p$ over some other field $\mathbb{F}_q$, resulting in significant overhead. For example, the implementation of secp256k1 and NIST P-256 ECDSA in the Cairo programming language (which uses an IOP-based STARK over a 254-bit, FFT-friendly, prime field $\mathbb{F}_q$) requires roughly 128 arithmetic operations over $\mathbb{F}_q$ to simulate a single $\mathbb{F}_p$ multiplication (this implementation uses various optimizations, the naive bit-wise multiplication would be far costlier).

Using the construction of Theorem 1.1 one may do better. The statement for each of these curves could be constructed over the native prime field $\mathbb{F}_p$, meaning that each multiplication gate in the computation of the ECDSA "costs" only one constraint, and addition comes for free. When computing the tradeoff between using an FFT-friendly field $\mathbb{F}_q$ or our new construction over $\mathbb{F}_p$, one should carefully measure the difference resulting from the new construction (which, as explained later, involves elliptic curves rather than plain polynomials). We leave this interesting question for future work, but speculate that in most cases the new $\mathbb{F}_p$-native constructions will be far better, in terms of prover time, verifier time, and proof length, than arithmetization over a different, yet FFT-friendly, field.

Next we discuss the four parts in which FFT-friendliness was demanded in prior scalable systems, and then explain how we get rid of this requirement.

[5]The rate parameter, defined as the ratio between a code's dimension and its blocklength, can be picked to be any constant $\rho < 1$, and affects the soundness error and proximity parameters; see [BCI+20] for state of the art soundness bounds as a function of rate.

1.2 Why do PCPs and IOPs require FFT-friendliness?

The very first step taken by a scalable PCP/IOP prover, when writing a proof for the integrity of a computation of length T, is to view the execution trace of the computation as a series of functions $f_1, \ldots, f_w : \mathsf{D} \to \mathbb{F}$ for some evaluation domain $\mathsf{D} \subset \mathbb{F}, |\mathsf{D}| = \mathsf{T}$, and then compute the low degree extension of each f_i by first interpolating the polynomial $P_i(X), \deg(P_i) < \mathsf{T}$ that agrees with f_i, and then evaluating $P_1, \ldots, P_w$ on a larger domain $\mathsf{D}' \subset \mathbb{F}, |\mathsf{D}'| \gg |\mathsf{D}|$, leading to a new sequence $f'_1, \ldots, f'_w : \mathsf{D}' \to \mathbb{F}$ that are submitted to the verifier as the very first part of the PCP/IOP. The four reasons D needs to be a cyclic group of size 2^{k} are explained next. If we wish to create scalable IOPs over all fields, including ones that do not contain such groups, we shall need to find other ways to achieve these properties.

- **Super-efficient Reed–Solomon encoding:** The main asymptotic bottleneck of scalable IOPs on the prover side is the computation of the low degree extensions of $f_1, \ldots, f_w$ from D to D'. When D is a subgroup of size 2^{k} and D' is a finite union of cosets of D, as used in all scalable PCP/IOP constructions, the classical FFT algorithm can be used to solve the encoding problem in time $O(w\mathsf{T}\log\mathsf{T})$; the asymptotic constants hidden by O-notation are rather small, which helps for concrete prover efficiency.
- **Codewords are invariant to cyclic shifts:** The algebraic constraints in AIRs over the trace involve elements from previous timesteps, which correspond to evaluations of $f'_1, \ldots, f'_w$ at translated arguments. Thus we need work not only with the codewords $f'_1, \ldots, f'_w$, but with words obtained by cyclic shifts of their values (where the cyclic order is determined by the indexing of the trace's elements by D). To control the degree of the evaluated constraints, it is necessary to know that these shifted words are also evaluations of polynomials of degree $< \mathsf{T}$, i.e. codewords. This is indeed the case when D is a cyclic group generated by g, D' is a finite union of its cosets, and the rows are indexed according to the cyclic order: shifting the values of $f'_i(x)$ by t yields the function $f'_i(\mathsf{g}^t x)$, which has the same degree as $f'_i(x)$ (each coset of D undergoes the same cyclic shift).
- **Polylogarithmic verification requires sparse domain polynomials:** To allow the verifier to check that the polynomial constraints arising out of the arithmetization reduction hold for each of the T steps of the computation, as claimed by the prover, the verifier needs to evaluate the "vanishing polynomial" of D, denoted $Z_{\mathsf{D}}(X)$, which is the degree-T monic polynomial whose roots are D, as well as polynomials that vanish on certain subsets $D_1, \ldots, D_s \subset \mathsf{D}$, denoted $Z_{D_i}(X)$. To facilitate scalable (polylogarithmic) verification, the verifier needs to evaluate $Z_{\mathsf{D}}(X), Z_{D_1}(X), \ldots, Z_{D_s}(X)$ all in time $\mathrm{polylog}\,\mathsf{T}$. When D is a multiplicative group of size T we have $Z_{\mathsf{D}}(X) = X^{\mathsf{T}} - 1$. This is a sparse polynomial that can be evaluated on any x_0 using $O(\log\mathsf{T})$ arithmetic operations. Likewise, when $D_1, \ldots, D_s$ are subgroups of D or, more generally, of "low-complexity" when expressed using subgroups (see Definition 2.3 for a definition of this term), then scalable (poly-logarithmic) verification is possible.
- **Low-degree testing:** Soundness of scalable PCPs/IOPs requires a protocol designed to verify that each of the functions $f'_1, \ldots, f'_w : \mathsf{D}' \to \mathbb{F}$ submitted by the prover is an RS codeword (or is close to it in Hamming distance). All scalable protocols — from the quasilinear RS-PCP of Proximity (PCPP) of [BS08] to the linear Fast RS IOP of Proximity (IOPP) protocol of [BBHR18] (abbreviated as FRI) — rely on the FFT-friendly structure of the domain D' over which functions are evaluated. In more detail, the fact that a cyclic group of size 2^{k} has a cyclic group of size $2^{\mathsf{k}-1}$ as a quotient group plays a vital role in the FRI protocol.

To summarize, there are four separate places in which FFT-friendliness is important in the construction of FRI-AIR STARK systems. RS encoding requires quasilinear running time over any finite field but the best asymptotic running time is obtained over multiplicative groups of order 2^{k}, i.e., within FFT-friendly fields. Expressing general constraints requires the RS codewords to be invariant to cyclic shifts, which occurs when the domain is itself a cyclic group. Scalable (poly-logarithmic) verification requires an evaluation domain that is represented by a sparse polynomial, and any multiplicative subgroup could be used. Finally, the low-degree testing protocol that lies at the heart of scalable PCP/IOP constructions requires an FFT-friendly domain.

1.3 Elliptic curves save the day, again

The virtues of elliptic curves in cryptography, computer science and mathematics are well established [Sil09, Was08, Kob87]. Here we make novel use of their properties — to create strictly scalable IOPs over any sufficiently large finite field, with the same asymptotic and concrete arithmetic complexity as obtained over FFT-friendly fields.

Our starting point is our recent work [BCKL21], that showed how to use elliptic curve groups to enable an FFT-like computation over all finite fields, thus enabling fast low degree extensions. This essentially gives us (with some small modifications) the analogue of the first item from Section 1.2. Developing analogues of the remaining three items is completely new to this paper, and it requires us to dig deeper than [BCKL21] into the elliptic curve group structure and properties of Riemann–Roch spaces over elliptic curves.

Another contribution of this paper is a randomized near-linear time algorithm for doing all the (one-time) precomputation required for the ECFFT and the EC based IOP. Additionally, in this paper we also provide a more explicit description of the curves and maps that appear in the isogeny chain, which in turn give us more explicit formulas for the FFTs themselves. This allows for easy implementation and easy determination of running time with concrete constants. See Section 4 and Section 4.3 in particular.

Taking a 30,000-feet view, fix any finite field $\mathbb{F}$ of size q. The family of elliptic curves defined over $\mathbb{F}$ is a family of algebraic groups whose size range and structure are well understood. Size-wise, nearly any number in the Hasse–Weil bound $[q+1\pm 2\sqrt{q}]$ is the size of some elliptic curve over $\mathbb{F}$ (when q is prime then *every* number in that range is the size of an elliptic curve). The group structure of elliptic curves is somewhat more elaborate, but suffice to say that for any size 2^{k}, there will exist some elliptic curve that contains a *cyclic*[6] subgroup of size 2^{k}, permitted that 2^{k} is, roughly, at most $\sqrt{q}$.

Based on these observations, we shall replace the multiplicative subgroup of size 2^{k} (which may not exist inside $\mathbb{F}_q^*$) with a cyclic subgroup of size 2^{k} of points of some elliptic curve E defined over $\mathbb{F}_q$. Then, we shall use a novel arithmetization scheme that reduces computational problems to problems regarding "low-degree" functions defined over the points of the elliptic curve; formally, these functions will be members of a low-degree Riemann–Roch (RR) space. The choice of this Riemann–Roch space in a way that enables arithmetization is the crux of our IOP construction, and we discuss this next.

1.3.1 Arithmetization and automorphisms

One property of polynomials (in the classical FFT-friendly field IOP setting) which is needed for efficient arithmetization is their invariance under certain linear transformations. In particular, if $G \subset \mathbb{F}_q$ is a multiplicative group generated by g, and $f : G \to \mathbb{F}_q$ is an evaluation of a polynomial of degree d, then $f(g \cdot x)$ is also a polynomial of degree d. In other words, the space of functions of degree at most d is invariant under the permutation that maps x to $g \cdot x$.

Now suppose we wish to arithmetize using a cyclic group H that is generated by a point h on an elliptic curve (i.e, H is a sub-group of the curve). A permutation that is natural in this context is given by $x \mapsto x + h$ (where x, h are points on the curve and $+$ is the curve's group operand). We need a space of functions that are invariant under this action, and this identifies a natural candidate space – the Riemann–Roch space of functions that is supported in a symmetric way on H, defined by the divisor $\sum_{z\in H}[z]$.

Another way of viewing this generalization is as follows. The space of polynomials of degree at most d in the projective space $\mathbb{P}^1$ (cf. Appendix A.1) is the Riemann–Roch space associated with the divisor $D = d \cdot [\infty]$ (see Eq. (30)), and D is invariant under the action $[x] \mapsto [g \cdot x]$. In the case of an elliptic curve group, $\infty \neq h + \infty$ so we cannot use D but rather need a different divisor, one that is invariant under the mapping induced by h. The natural divisor is $D' := \sum_{z\in H}[z]$ which is clearly invariant under the action of h because H is cyclic.

[6] The need for cyclic subgroups of size 2^{k}, as opposed to general subgroups of size 2^{k}, of elliptic curve groups is new to this paper in comparison to [BCKL21]. The cyclicity is essential for arithmetization.

1.3.2 Key ingredients for the new IOPs, and the relationship to ECFFT Part I [BCKL21]

Let us now see the elliptic curve analogues of the four ingredients that go into IOPs in FFT-friendly fields. The first of these essentially comes from [BCKL21].

- **Super-efficient EC code encoding:** This essentially comes from [BCKL21]. Here we generalize the results slightly to extend low-degree functions evaluated over D to evaluations over a constant number of other cosets of D, in time $O(\mathsf{T} \log \mathsf{T})$ and with small concrete asymptotic constants. See Section 6.2 for details.
- **Invariance to cyclic shifts:** This is where the choice of the Riemann–Roch space is crucially used. It was specifically constructed to be invariant to translation of the argument by any element of the cyclic subgroup of size 2^{k} in E, similarly to the case of polynomials with bounded degree. Since D' is a union of cosets of the cyclic subgroup, these translations correspond to cyclic permutations of each coset in D'. See Section 5.4 for details.
- **Polylogarithmic evaluation of the "vanishing RR function" of D:** The verifier now needs to evaluate "low-degree" "vanishing RR functions" (the analogue of a vanishing polynomial in the Riemann–Roch space) $\hat{Z}_{\mathsf{D}}(P)$ on an arbitrary point $P = (x_0, y_0)$ of E, where $\hat{Z}_{\mathsf{D}}$ is the RR function that vanishes over D. It turns out that D can be constructed using a sequence of $k = \log \mathsf{T}$ rational functions and this implies that $\hat{Z}_{\mathsf{D}}(P)$ is computable using $O(\log \mathsf{T})$ arithmetic operations, as before. Likewise, for subsets $D_1, \ldots, D_s \subset \mathsf{D}$ of "low complexity" (per Definition 2.3), the verifier can evaluate $\hat{Z}_{D_i}(P)$ as efficiently for subsets of elliptic curves as was the case with subsets of multiplicative groups. See section Section 7 for details.
- **Low-degree testing:** The FRI protocol can be described informally as "random folding of an FFT". Thus, once we have obtained a generalization of the FFT algorithm to codes defined over elliptic curve groups, we also generalize the FRI protocol to verify the proximity of functions to low-degree RR functions. Details appear in Section 6.3.

1.4 Related work

Over the past decade we have experienced a Cambrian explosion in the field of concretely efficient proof systems, with and without zero knowledge. These systems are classified under various definitions including CS proofs [Mic00], NIZKs and succinct NIZKs [GGPR13], SNARGs, SNARKs, STARKs, and more. Realizations in code include Pinocchio [PGHR13], C-SNARKs [BCG+13], PLONK [GWC19], Halo [BGH19], Fractal [COS20], Marlin [CHM+20], Ligero [AHIV17], Sonic [MBKM19], Bulletproofs [BBB+18] and more.

Nearly all of these systems involve arithmetization via polynomials (univariate and multivariate) over large fields, of size at least $\mathsf{poly}(\mathsf{T})$, and thus when efficiency (concrete and asymptotic) is of interest, FFT-friendliness is required, along with proving time that is quasi-linear (or worse). An interesting research question, not addressed here, is whether the techniques discussed in this paper are relevant to some of these works. It seems likely to conjecture that many of the works that are information theoretically secure, like the important lines of works based on "interactive proofs for muggles" [GKR08] and "MPC in the head" [IKOS07] may be constructed with better efficiency over general large fields, using our results.

A class of concretely efficient and widely deployed ZK-SNARK systems is based on knowledge-of-exponent assumptions and bi-linear pairings, starting with the work of [PGHR13]. Several blockchain systems, including Zcash, Filecoin and Tornado cash use the popular and efficient Groth16 ZK-SNARK [Gro16]. The use of bilinear pairings significantly limits the class of fields that can be arithmetized efficiently, requiring $\mathbb{F}$ to be a prime field with small embedding degree and ruling out fields that are of prime power size[7]. Other constructions that rely on number-theoretic assumptions but which do not require knowledge of exponent assumptions, nor bilinear pairings (e.g., BulletProofs

[7] Arithmetization in the context of such SNARKs has as its output a system of R1CS constraints defined over an elliptic curve subgroup of prime order p that has small constant embedding degree.

and Halo), may be amenable to efficient constructions over non-FFT friendly, cryptographically large primes/curves (but it seems unlikely they can be amended to allow native arithmetization over fields of small characteristic).

An interesting and noteworthy recent line of works gives strictly linear proving time, thereby avoiding the need for FFTs [BCG+17b, BCG20, RR21, **?**] and large fields and offering strictly better asymptotic proving time than mentioned above. However, thus far this line of works has not produced scalable systems (per the definition above) and requires super-polylogarithmic verification time which should be performed either directly by the verifier or by a preprocessing entity trusted by it. In particular, our main results (Theorems 1.1 and 1.4) do not imply these works and vice versa.

Elliptic curves and FFT. This work is a direct continuation of our previous paper on quasilinear time Elliptic Curve FFT [BCKL21] (cf. [CC89] for an earlier work on using elliptic curves to compute an FFT-like transform, as well as the discussion in [BCKL21] of that paper). Indeed, the sequence of isogenies used in Section 4.2 is adapted from that work, and the EC-FRI protocol of Section 6.3 relies on our FFT-like interpolation and evaluation algorithms of that work. Although we made this paper self-contained, reading our previous work should help the reader with intuition (and notation) here. See Section 1.3 for a detailed discussion of what is new in this paper in comparison to [BCKL21].

Algebraic Geometry codes and PCPs/IOPs A line of works used algebraic geometry codes to obtain PCPs and IOPs with extremely efficient proof length and query complexity over constant size fields [BKK+16, BCG+17a]. Those works are incomparable to ours because the curves there are of much higher genus, and the end results are not related to our goal of constructing scalable proof systems over any finite field.

1.5 Outline of the rest of the paper

The next section presents the main results more formally, and Section 3 gives a self-contained proof of our main result (Section 3). This proof relies on further results discussed later, as follows. Section 4 gives an efficient procedure for selecting a "good", i.e., "FFT-like" sequence of curves, needed for the IOP. Section 5 defines the family of elliptic curve codes used by our arithmetization, and discusses some of their properties. Section 6 presents quasi-linear algorithms for encoding (or "low-degree extending") functions in this family of codes, and analyzes efficient IOPs of proximity for them. Section 7 defines and discusses the zero loci of elliptic curve code members that is needed to define the enforcement domains of AIR constraints. The appendix contains mathematical definitions and a more detailed analysis of the EC version of the FRI protocol (Appendices A and B, respectively).

We note that Sections 4 to 7 use notations which differ slightly from the notation used in Section 3. The main reason for this is that the in-depth sections must deal not with a single curve, Riemann–Roch space etc., but with families and chains of such, and the relationships between them, due to the recursive nature of algorithms such as FFT and FRI, and the likewise recursive structure of this objects. These families require an additional level of indexation. However, to simply use FFT and FRI as done in Section 3, it is not necessary to go beyond the first layer. Thus, we are able to use a slightly simplified notation in that section.

2 Main results

Our main result below is a scalable and transparent IOP of knowledge (abbreviated as STIK) for the language of satisfiable AIR instances defined over *any* sufficiently large finite field. Thus, we start by defining this language (Definition 2.6). Then we state and discuss our main theorem (Theorem 2.10). We conclude with a statement of the auxiliary results on FRI and EC-FRI over any finite field.

2.1 The AIR Language and Relation

We recall the definition of an AIR instance from [BBHR19], using the more recent formulation in [Sta21, Section 5], generalizing it slightly by using an abstract cyclic group instead of a multiplicative group[8] of a finite field. As shown in that paper, this language, even when restricted to FFT-friendly fields, is NEXP-complete. We start with the notion of an AIR instance.

Definition 2.1 (AIR Instance)**.** *An Algebraic Intermediate Representation (AIR) instance is a tuple* $\mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{I}, \mathsf{Cset})$ *where:*

- $\mathbb{F}$ *is a finite field*
- $\mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}$ *are integers indicating the following sizes:*
  - w *is the number of columns in the trace*
  - h *denotes the logarithm of the size of the trace domain*
  - d *is the maximal degree of a constraint*
  - s *is the size of the set of constraints*
- H_0 *is a cyclic group of size* 2^{h}*, and* g *is a generator of it. We write* H_0 *multiplicatively, so that* $\mathsf{g}^j \cdot y$ *means applying* g^j *(the* j*-length cyclic shift) to* y*. We call* H_0 *the* trace domain.
- $\mathsf{I} \subseteq \{0, 1, \ldots, 2^{\mathsf{h}} - 1\} \times \{1, \ldots, \mathsf{w}\}$ *is a set of pairs known as the set of* mask indices*. Let* $\mathsf{Z} = \{\mathsf{Z}_{j,l} : (j, l) \in \mathsf{I}\}$ *be a set of formal variables, called the* mask variables*, indexed by elements of* I*.*
- $\mathsf{Cset} = \{\mathsf{C}_1, \ldots, \mathsf{C}_{\mathsf{s}}\}$ *is a finite set of constraints, of size* s*. Each constraint is an ordered pair* $\mathsf{C}_\alpha = (\mathsf{Q}_\alpha, \mathsf{H}_\alpha)$ *where:*
  - $\mathsf{Q}_\alpha \in \mathbb{F}^{\leq \mathsf{d}}[\mathsf{Z}]$ *is a multivariate polynomial over the mask variables, of total degree at most* d*, called the* α-th constraint polynomial*.*
  - $\mathsf{H}_\alpha \subseteq \mathsf{H}_0$ *is a subset of the group, called the* α-th constraint enforcement domain*.*

The kind of result we will show is that the language of satisfiable AIRs over every field has an efficient IOPP. The efficiency will be in terms of the complexity of the constraints of the AIR, which we define next. Informally, the complexity of the AIR constraints depend on two things. The first is the circuit complexity of individual constraints, defined first (Definition 2.2). The second, less trivial, component, is the specification of the domain on which different constraints must be enforced (Definition 2.3).

Definition 2.2 (Complexity of Constraints of an AIR)**.** *Given an AIR* $\mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{I}, \mathsf{Cset})$*, we define the complexity of the constraints of* A*, denoted* $\|\mathsf{Cset}\|$*, as:*

$$\|\mathsf{Cset}\| := \sum_{\alpha=1}^{\mathsf{s}} (\|\mathsf{Q}_\alpha\| + \|\mathsf{H}_\alpha\|),$$

where $\|\mathsf{Q}_\alpha\|$ *is the arithmetic complexity of the circuit computing the polynomial* Q_α*, and* $\|\mathsf{H}_\alpha\|$ *is the coset complexity of* H_α *(see definition below).*

As motivation for the following definition, consider a linear computation in which a constraint should be applied only to half of the timesteps. Informally, a constraint applied periodically, every other step (on even-numbered time steps) has lower complexity than a constraint that should be applied to a randomly selected set of time steps. We define the set of relevant time steps using polynomials and rational functions, and it turns out the the following measure is an upper bound on their complexity as arithmetic circuits.

[8] An AIR can also be defined using Hamiltonian paths in affine graphs, but restricting to cyclic groups suffices for NEXP-completeness, see [BBHR19].

Definition 2.3 (Coset Complexity)**.** *For a subset S of a finite group H, we define the* coset complexity *of S, denoted $\|S\|$, to be the smallest value of*

$$\sum_i (\log_2(|J_i|) + 1),$$

over all ways of writing the indicator function $\mathbf{1}_S$ of S as a signed sum of indicator functions:

$$\mathbf{1}_S = \sum_i \epsilon_i \cdot \mathbf{1}_{J_i},$$

where each J_i is a coset of a subgroup of H and $\epsilon_i = \pm 1$.

Next, we recall the definition of an AIR witness.

Definition 2.4 (AIR witness and composition)**.** *An* AIR witness *is a sequence of functions $\vec{\mathsf{f}} = (\mathsf{f}_1, \ldots, \mathsf{f}_\mathsf{w})$, where each f_l is a function from H_0 to $\mathbb{F}$. The* witness size *is $\mathsf{w} \cdot |\mathsf{H}_0|$.*

Given an AIR constraint polynomial $\mathsf{Q} \in \mathbb{F}[\mathsf{Z}]$, the composition *of Q and the witness $\vec{\mathsf{f}}$ is the function*

$$\mathsf{Q} \circ \vec{\mathsf{f}} : \mathsf{H}_0 \to \mathbb{F},$$

where, for all $y \in \mathsf{H}_0$:

$$(\mathsf{Q} \circ \vec{\mathsf{f}})(y) = \mathsf{Q}\left(\left(\mathsf{f}_l(\mathsf{g}^j \cdot y)\right)_{j,l}\right).$$

(On the right hand side, we replaced the variable $\mathsf{Z}_{j,l} \in \mathsf{Z}$ that appears in $\mathsf{Q}(\mathsf{Z})$ with $\mathsf{f}_l(\mathsf{g}^j \cdot y)$).

We now define which witnesses are said to satisfy an instance. As motivation, consider a typical way that an AIR can encode a computation. We could have a machine with w $\mathbb{F}_q$-registers, and ask that $\mathsf{f}_l(g^j)$ represents the contents of the l-th register at time j. Then we use the constraints to (1) capture the transition rules between time step j and $j+1$ for all j in the enforcement domain $[0, T]$, and (2) enforce boundary constraints on the values of the registers at time 0 and at time T.

Definition 2.5 (Satisfiability)**.** *We say that the AIR witness $\vec{\mathsf{f}} = (\mathsf{f}_1, \ldots, \mathsf{f}_\mathsf{w})$* satisfies *the AIR instance $\mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{I}, \mathsf{Cset})$ if and only if*

$$\forall \alpha \in [\mathsf{s}]: \quad y \in \mathsf{H}_\alpha \Rightarrow (\mathsf{Q}_\alpha \circ \vec{\mathsf{f}})(y) = 0.$$

In words, $\vec{\mathsf{f}}$ satisfies A iff for every constraint $\mathsf{C}_\alpha = (\mathsf{Q}_\alpha, \mathsf{H}_\alpha) \in \mathsf{Cset}$ it holds that $\mathsf{Q}_\alpha \circ \vec{\mathsf{f}}$ vanishes on the α-th constraint enforcement domain H_α. We say that the AIR A is satisfiable if there exists an AIR witness $\vec{\mathsf{f}}$ that satisfies it.

We now reach the main definition of this subsection, that of the language, and relation, corresponding to satisfiable AIRs over fields of quadratic size.

Definition 2.6 (AIR Language/Relation)**.** *The AIR relation R_AIR is*

$$\begin{aligned}\mathsf{R}_\mathsf{AIR} = \{(\mathsf{A}, \vec{\mathsf{f}}) \mid \mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{I}, \mathsf{Cset}) \text{ is an AIR},\\ \vec{\mathsf{f}} \text{ is a satisfying AIR witness for } \mathsf{A},\\ |\mathbb{F}| \geq \Omega(\mathsf{d}^2 \cdot 2^{2\mathsf{h}})\}.\end{aligned}$$

The language of satisfiable AIRs is the projection of R_AIR onto its first coordinate,

$$\mathsf{L}_\mathsf{AIR} = \{\mathsf{A} \mid \exists \vec{\mathsf{f}}\, (\mathsf{A}, \vec{\mathsf{f}}) \in \mathsf{R}_\mathsf{AIR}\}.$$

Remark 2.7 (Field size). The definition above requires $|\mathbb{F}| > (\mathsf{d}|\mathsf{H}_0|)^2$. When this is not the case one may embed $\mathbb{F}$ is a finite extension field $\mathbb{K}$ that is sufficiently large and apply our results to the AIR over $\mathbb{K}$. This increases the various complexity measures (proving time, verification time and query complexity) by a multiplicative factor of at most $M([\mathbb{K} : \mathbb{F}])$, where $M([\mathbb{K} : \mathbb{F}])$ denotes the complexity of $\mathbb{K}$-multiplication in terms of arithmetic operations over $\mathbb{F}$; notice that $M(k) \leq k^2$ for any $\mathbb{K}$ that is the degree k extension of $\mathbb{F}$. For instance, in the extremal case of the smallest possible field size, $\mathbb{F}_2$, any AIR per Definition 2.1 over $\mathbb{F}_2$, using an (abstract) group H_0 of size n, would lead to using $k = 2\log n + O(1)$, leading to total prover complexity of $O(n \log n \cdot M([\mathbb{F}_{2^{2\log n + O(1)}} : \mathbb{F}_2]) \leq O(n \log^3 n)$ measured in arithmetic operations over $\mathbb{F}_2$.

2.2 A Scalable and Transparent IOP for $\mathsf{L}_{\mathsf{AIR}}$

To state our main result we assume familiarity with the definition of an IOP, and briefly recall its main parameters [BCS16, RRR16].

An Interactive Oracle Proof (IOP) for a language L is an interactive proof system defined by a prover P and verifier V, in which the verifier need not read the prover's messages in full. Rather, the IOP model allows the verifier oracle access to the prover's messages. (The prover is assumed to read all verifier messages in entirety.) The main parameters of interest are:

- *query complexity* q is the total number of symbols queried by the verifier from the prover's messages
- *round complexity* k is the number of rounds of interaction between the two parties.
- *prover complexity* $\mathsf{time}_{\mathsf{P}}$ and verifier complexity $\mathsf{time}_{\mathsf{V}}$, which, in this paper, will assume unit cost for arithmetic operations over the ambient field
- *proof length* l is the sum of lengths of oracles sent by the prover throughout the protocol.
- *soundness error* err is the probability of the verifier accepting a false statement.

Main Result. It was shown by [BBHR19] that the sub-language of $\mathsf{L}_{\mathsf{AIR}}$ restricted to FFT-friendly fields has a scalable and transparent IOP of knowledge. Formally, let

$$\mathsf{L}_{\mathsf{AIR},\mathsf{FFT}} = \{\mathsf{A} \in \mathsf{L}_{\mathsf{AIR}} \mid \mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{l}, \mathsf{Cset}) \text{ satisfies } 2^{\mathsf{h}} \mid |\mathbb{F}| - 1\}.$$

The main theorem of [BBHR19] is:

Theorem 2.8 (STIK for $\mathsf{L}_{\mathsf{AIR},\mathsf{FFT}}$ – Prior state of art)**.** *There is an IOP protocol for the language* $\mathsf{L}_{\mathsf{AIR},\mathsf{FFT}}$ *such that for* $\mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{l}, \mathsf{Cset})$ *of witness size* $n = \mathsf{w} \cdot 2^{\mathsf{h}}$ *and parameter* t *we have:*

- **Completeness, Proving time and Proof size:** *There is a Prover algorithm that given* $\vec{\mathsf{f}}$ *such that* $(\mathsf{A}, \vec{\mathsf{f}}) \in \mathsf{R}_{\mathsf{AIR}}$*, makes the verifier accept with probability* 1*. Prover running time is*

  $$O(n \cdot (\log n + \|\mathsf{Cset}\|)),$$

  and proof length l *is* $O(n)$*.*
- **Verifier runtime and query complexity:** *For all* $\mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{l}, \mathsf{Cset})$*, the verifier runs in time*

  $$O(\|\mathsf{Cset}\| + t \cdot \mathsf{h}).$$

  and makes a total of of $\mathsf{q} \leq t \log n$ *queries*
- **Knowledge soundness and soundness:** *There exists an efficient extractor running in time* $\mathsf{poly}(n)$ *such that, given access to a Prover which satisfies the verifier with probability greater than* 2^{-t}*, outputs* $\vec{\mathsf{f}}$ *such that* $(\mathsf{A}, \vec{\mathsf{f}}) \in \mathsf{R}_{\mathsf{AIR}}$*. In particular, if* $\mathsf{A} \notin \mathsf{L}_{\mathsf{AIR}}$ *then, for any Prover strategy, the verifier will reject with probability at least* $1 - 2^{-t}$*.*

Remark 2.9 (Soundness and knowledge soundness). Often in the analysis of interactive proofs, the soundness error parameter is smaller than the soundness parameter. In the theorem above we state the same parameter for both because the state-of-the-art soundness analysis in our case is actually efficient, and uses a witness extractor.

The first step of the above IOPP is to identify the cyclic group H_0 with a subgroup of the multiplicative group $\mathbb{F}_q^*$, and to view satisfying AIR witnesses $\mathsf{f}_l : \mathsf{H}_0 \to \mathbb{F}_q$ as the values of a low degree univariate polynomial $\mathsf{f}_l(Y) \in \mathbb{F}_q[Y]$. This then makes the AIR a collection of constraints on the values of low-degree polynomials at certain points of the field $\mathbb{F}_q$, and brings the tools of algebra into play.

The FFT-friendliness is crucial for this approach — without it, there is no suitable multiplicative subgroup in $\mathbb{F}_q^*$ to identify the cyclic group H_0 with, and the above approach fails to get off the ground (see Section 1.2).

Our main result, given below, removes the FFT-friendliness restriction, and gives an IOPP for satisfiable AIRs over *all* finite fields with almost identical guarantees as Theorem 2.8. The key ingredient is to identify the cyclic group H_0 with a cyclic subgroup of an elliptic curve E over $\mathbb{F}$, and to view satisfying AIR witnesses $\mathsf{f}_l : \mathsf{H}_0 \to \mathbb{F}_q$ as the values of low degree rational functions f_l defined[9] on the curve E.

Theorem 2.10 (Scalable and Transparent IOPs of Knowledge over all large fields)**.** *There is an IOP protocol for the language $\mathsf{L}_{\mathsf{AIR}}$ with properties and parameters as stated in Theorem 2.8 above.*

The complexity parameters of the theorem, along with completeness, are argued along the lines of the proof of Theorem 2.8 (see [Sta21, Section 5]). The most delicate part is the soundness analysis (as is always the case with IOP systems). The proof appears in Section 3.3.

EC-STARKs Assuming the existence of a family of collision resistant hash functions, and replacing the interactive oracles with Merkle commitment schemes a la [Kil92], one obtains an interactive Scalable Transparent ARgument of Knowledge (STARK) as defined in [BBHR19]. Alternatively, working in the random oracle model and applying the BCS reduction [BCS16], one obtains a noninteractive STARK (which is also, in particular, a transparent SNARK). Details of both reductions are identical to prior STARKs and discussed elsewhere (e.g., [Kil92, Mic00, BCS16, CY21b, CY21a]). We point out that STARKs based on FFT-friendly fields (Theorem 2.8) are concretely practical, as evidenced by the StarkEx system which implements them to scale transactions on Ethereum. We conjecture that the new EC-based construction of Theorem 1.1 will have practical applications in certain settings (as discussed in Section 1.1).

2.3 IOPs of Proximity (IOPPs) for RS codes over all large fields

In this section we state our auxiliary main result: FRI over all large finite fields. We start with a few necessary definitions.

We use Δ to denote relative Hamming distance between two vectors $u, v \in \mathbb{F}^n$, defined as $\Delta(u,v) = \frac{1}{n}|\{i \in [n] \mid u_i \neq v_i\}|$, and for a set $V \subset \mathbb{F}^n$ we let $\Delta(u,V) = \min\{\Delta(u,v) \mid v \in V\}$. The agreement of u, v and u, V is defined to be $\mathsf{agree}(u,v) = 1 - \Delta(u,v), \mathsf{agree}(u,V) = 1 - \Delta(u,V)$.

Definition 2.11 (IOP of Proximity (IOPP))**.** *Fix $V \subset \mathbb{F}^n$. An IOP system (P, V) is said to be an IOP of proximity (IOPP) for V with soundness error function $\mathsf{err} : [0,1] \to [0,1]$ (and additional complexity parameters as defined for standard IOP systems above) if, assuming the verifier has oracle access to $v \in \mathbb{F}^n$, the following hold:*

- *There exists a prover P such that for $v \in V$,*

$$\Pr\left[\langle \mathsf{V}^v \leftrightarrow \mathsf{P}(v)\rangle = \mathsf{accept}\right] = 1$$

- *If $v \notin V$ (so $\Delta(v,V) > 0$) then for any prover P^* we have*

$$\Pr\left[\langle \mathsf{V}^v \leftrightarrow \mathsf{P}(v)\rangle = \mathsf{accept}\right] \leq \mathsf{err}(\Delta(v,V))$$

[9] To be precise, we work with a suitable Riemann–Roch space.

Reed Solomon Codes Let $\mathsf{RS}[\mathbb{F}_q, L, \rho]$ denote the Reed–Solomon code over field $\mathbb{F}_q$, evaluation domain L and rate ρ:

$$\mathsf{RS}[\mathbb{F}_q, L, \rho] = \{f : L \to \mathbb{F}_q : \deg(f) < \rho|L|\}. \tag{1}$$

Recall the previous state of the art with respect to IOPPs for Reed–Solomon codes. We call a finite field $\mathbb{F}$ *n-smooth* if it contains a sub-group (additive or multiplicative) of size $n = 2^{\mathsf{k}}$ for integer k.

Theorem 2.12 (FRI over smooth fields [BBHR18, BCI+20])**.** *Let $\mathbb{F}$ be an n-smooth finite field. Then there is a subset $L \subseteq \mathbb{F}$ with size n such that for any rate parameter $\rho = 2^{-\mathcal{R}}$ ($\mathcal{R} \in \mathbb{N}$) and repetition parameter t, the Reed–Solomon code $\mathsf{RS}[\mathbb{F}, L, \rho]$ has an IOPP with:*

- *linear proving time $\mathsf{time_P} = O(n)$ and proof length $\mathsf{l} < n$,*
- *logarithmic query complexity $\mathsf{q} = t \cdot \log(n) + O(1)$ and verification time $\mathsf{time_V} = O(t \log n)$*
- *soundness error function err, where:*

$$\mathsf{err}(\delta) = O\left(\frac{n^2}{q}\right) + \left(\min(\delta, 1 - \sqrt{\rho}) - o(1)\right)^t.$$

Our second main result shows essentially the same bounds over any finite field, not just smooth ones.

Theorem 2.13 (FRI over all fields)**.** *Let $\mathbb{F}$ be the finite field of size q, a prime power. Then for every $n \leq O(\sqrt{q})$ there exists a set $L \subseteq \mathbb{F}_q$ of size $\Theta(n)$ such that for any rate parameter $\rho = 2^{-\mathcal{R}}$ ($\mathcal{R} \in \mathbb{N}$) and repetition parameter t the Reed–Solomon code $\mathsf{RS}[\mathbb{F}, L, \rho]$ has an IOPP with the complexity measures as stated in Theorem 2.12.*

2.4 Fast IOPs of Proximity for Elliptic Curve Codes

We generalize Theorem 2.13 to certain algebraic geometry codes, evaluations of functions in a low-degree Riemann–Roch space over FFT-friendly subgroups of elliptic curves (definitions of these terms appear in Appendix A). To define the specific codes recall the definition of Algebraic Geometry (or Goppa) codes.

Definition 2.14 (Algebraic Geometry Codes)**.** *Let X be a non-singular projective curve over a field $\mathbb{F}$, let $D = \{x_1, \ldots, x_n\}$ be a set of $\mathbb{F}$-rational points and G be a divisor with support disjoint from D. Let $\mathscr{L}(G)$ be the Riemann–Roch space defined by G. Then the* algebraic geometry (AG) code *(also known as a Goppa code) $C(D, G)$ is*

$$C(D, G) := \{f(x_1), \ldots, f(x_n) \mid f \in \mathscr{L}(G), x_i \in D\} \tag{2}$$

Our next result is the following.

Theorem 2.15 (Fast Elliptic Curve Code IOPP)**.** *Let E be an elliptic curve over $\mathbb{F}$, let $G \subset E$ be a cyclic group of size 2^{h} and let D be a union of m nontrivial and disjoint cosets of G, such that $G \cap D = \emptyset$. Let $[G] := \sum_{P \in G}[P]$ be the divisor naturally associated with G (see Appendix A.5). Then, for any repetition parameter t and setting $\rho = 1/m$, the AG code $C(D, [G])$ has an IOPP with complexity parameters as in Theorem 2.12.*

3 Scalable IOPs for AIRs over any large field

In this section we prove our main theorem – Theorem 2.10, relying on certain claims that are proved in later sections.

3.1 The ECFFT Infrastructure

The proof of Theorem 2.10 relies on delicately chosen elliptic curves, subgroups of those curves, Riemann–Roch spaces and AG codes, and special "degree-correction" functions on the curve. All of these are explained meticulously, and the required properties proven formally, in later sections. The goal of this section is to lay out, in a self-contained manner, all the results which are needed to derive our main results regarding IOPs and IOPs of proximity (in Section 3.2).

This section builds upon our recent results in [BCKL21], and we advise the reader to consult that paper regarding the results quoted here from that work.

3.1.1 The EC backbone

The backbone of all of the constructions in this paper is the chain of 2-isogenies whose existence was shown in [BCKL21, Theorem 4.9], which we quote here:

Theorem 3.1 (Good Curve Sequence)**.** *For any prime power $q \geq 7$ and any $1 < \mathsf{K} = 2^{\mathsf{k}} \leq 2\sqrt{q}$, there exist elliptic curves $E_0, E_1, \ldots, E_{\mathsf{k}}$ over $\mathbb{F}_q$ in extended Weierstrass form, a subgroup $G_0 \subseteq E_0$ of size K, 2-isogenies $\varphi_i : E_i \to E_{i+1}$ and rational functions $\psi_i : \mathbb{P}^1 \to \mathbb{P}^1$ of degree 2, such that the following diagram is commutative:*

$$\begin{array}{ccccccc} E_0 & \xrightarrow{\varphi_0} & E_1 & \xrightarrow{\varphi_1} & \cdots & \xrightarrow{\varphi_{\mathsf{k}-1}} & E_{\mathsf{k}} \\ \downarrow{\scriptstyle \pi_0} & & \downarrow{\scriptstyle \pi_1} & & & & \downarrow{\scriptstyle \pi_{\mathsf{k}}} \\ \mathbb{P}^1 & \xrightarrow{\psi_0} & \mathbb{P}^1 & \xrightarrow{\psi_1} & \cdots & \xrightarrow{\psi_{\mathsf{k}-1}} & \mathbb{P}^1 \end{array} \tag{3}$$

where:

- *π_i are the projection maps to the x-coordinate of each curve;*
- *$|\varphi_{i-1} \circ \cdots \circ \varphi_0(G_0)| = \frac{1}{2^i}|G_0| = 2^{\mathsf{k}-i}$.*
- *G_0 has a coset C such that $C \neq -C$ (as elements of the quotient group $E_0(\mathbb{F}_q)/G_0$).*

Note that this theorem is very abstract: It only establishes the existence of these curves and maps, but says almost nothing about the form of the equations defining E_i or of the isogenies φ_i and maps ψ_i, does not specify the structure of G_0, and does not show how to find such curves.

In Section 4 we revisit this theorem, and strengthen and refine it for our needs. First, we show a realization of the above curve sequence using elliptic curves E_i of a simple form, and and obtain simple, explicit formulas for φ_i and ψ_i. Next, we show how to get the above sequence with G_0 being a *cyclic* group (isomorphic to $\mathbb{Z}/2^{\mathsf{k}}\mathbb{Z}$) — this is crucial for doing efficient arithmetization of AIRs (which are defined in terms of cyclic groups). Finally, we give a probabilistic algorithm for finding such curves in nearly optimal $O(2^{\mathsf{k}} \operatorname{polylog} q)$ time. The following statement summarizes these improvements to Theorem 3.1.

Theorem 3.2. *There exists a randomized algorithm* Find Curve*, that on input* k *and $q \geq \max\left\{7, 2^{2(\mathsf{k}-1)}\right\}$, runs in time $O(2^{\mathsf{k}} \log^2 q \log\log q)$, and with high probability finds elliptic curves E_i in Weierstrass form and maps φ_i, ψ_i as in Theorem 3.1, such that G_0 is a* **cyclic** *group of size 2^{k} and the maps φ_i, ψ_i are computable via $O(1)$ operations in $\mathbb{F}_q$.*

The upper bound on the algorithm's runtime can be improved by a $\tilde{O}(\log q)$ factor assuming the Riemann Hypothesis, and we believe that it should be even faster. For details see Section 4.

3.1.2 Function Spaces and Evaluation Domains

We are now ready to explicitly describe the setup we will need for our IOP for satisfiable AIRs. For analogues of the FFT and IFFT algorithms and the FRI protocol, we will need to identify some special functions and some special sets of evaluation points. These are captured below.

Setup 3.3. *For every q, k with $q \geq \Omega(2^{2\mathsf{k}})$, there exists an elliptic curve $\mathsf{E}/\mathbb{F}_q$ such that $\mathsf{E}(\mathbb{F}_q)$ contains a cyclic group G of size 2^{k}.*

Fixing such a curve E, we introduce some notation:

- *For each $\ell \leq \mathsf{k}$, let $G^{\langle \ell \rangle}$ be the cyclic subgroup of G of size 2^ℓ.*
- *A* basic subset *S of $\mathsf{E}(\mathbb{F}_q)$ at scale ℓ is a set $S = C \cup (-C)$, where $C \subseteq \mathsf{E}(\mathbb{F}_q)$ is a coset of $G^{\langle \ell \rangle}$ with $C \neq -C$. Note that $|S| = 2|C| = 2^{\ell+1}$.*
- *An* evaluation domain *$\mathbf{S}$ of $\mathsf{E}(\mathbb{F}_q)$ at scale ℓ is a union of disjoint basic subsets of $\mathsf{E}(\mathbb{F}_q)$ at scale ℓ.*
- *Let $\mathcal{K}^{\langle \ell \rangle}$ be the $\mathbb{F}_q$-linear space $\mathscr{L}([G^{\langle \ell+1 \rangle}])$ of rational functions on E. By the Riemann–Roch theorem, we have $\dim(\mathcal{K}^{\langle \ell \rangle}) = 2^{\ell+1}$.*

We now set up similar notions on the projective line, obtained by projecting down to the x-coordinate via the map π. The curve E is assumed to be in Weierstrass form.

- *A* basic subset *T of $\mathbb{F}_q$ at scale ℓ is the projection $T = \pi(S)$ of a basic subset of $\mathsf{E}(\mathbb{F}_q)$ at scale ℓ. Note that $|T| = 2^\ell$.*
- *An* evaluation domain *of $\mathbb{F}_q$ at scale ℓ is a union of disjoint basic subsets of $\mathbb{F}_q$ at scale ℓ. Equivalently, it is a set of the form $\mathbf{T} = \pi(\mathbf{S})$, where $\mathbf{S}$ is an evaluation domain of $\mathsf{E}(\mathbb{F}_q)$.*
- *Let $\mathcal{M}^{\langle \ell \rangle}$ denote the space of polynomials in $\mathbb{F}_q[X]$ of degree at most $2^\ell - 1$. Note that $\dim(\mathcal{M}^{\langle \ell \rangle}) = 2^\ell$.*

The $\mathcal{K}^{\langle \ell \rangle}$ and $\mathcal{M}^{\langle \ell \rangle}$ spaces above are related through a certain univariate polynomial $\Omega^{\langle \ell \rangle}(X)$ of degree exactly $2^\ell - 1$ (see Section 5.1 for an explicit description). Corollary 5.9 shows that every rational function $f(X, Y) \in \mathcal{K}^{\langle \ell \rangle}$ can be written uniquely in the following form:

$$f(X, Y) = \frac{1}{\Omega^{\langle \ell \rangle}(X)} \left(f_0(X) + \frac{Y}{X} f_1(X) \right), \tag{4}$$

where $f_0(X), f_1(X) \in \mathcal{M}^{\langle \ell \rangle}$. We will sometimes write this as:

$$f(Z) = \frac{1}{\Omega^{\langle \ell \rangle}(\pi(Z))} \left(f_0(\pi(Z) + \zeta(Z) f_1(\pi(Z)) \right),$$

where $Z = (X, Y)$ is a pair of formal (related) variables representing a point on the curve, π is the projection from E onto the x-coordinate, and $\zeta((X, Y)) = \frac{Y}{X}$.

This representation will let us move between the space of rational functions $\mathcal{K}^{\langle \ell \rangle}$ and the space of polynomials $\mathcal{M}^{\langle \ell \rangle}$.

3.1.3 FFT and IFFT

The following theorems give the new FFT and IFFT transformations that we will need. The proofs of the following theorems appear in Section 6.1. The bases that appear in the theorems are defined in Definitions 5.5 and 5.8. Following the notation in [BCKL21], for a function f defined on an evaluation domain S, we denote by $\langle f \wr S \rangle$ the *evaluation table* of f on S. When f belongs in a linear space spanned by a basis β, we denote by $[f]_\beta$ the representation of f in the basis.

Theorem 3.4 (FFT and IFFT- Elliptic Curve Version)**.** *For each ℓ, there is a basis $\kappa^{\langle \ell \rangle} = (\kappa_j^{\langle \ell \rangle})_{j=0}^{2^{\ell+1}-1}$ of $\mathcal{K}^{\langle \ell \rangle}$ such that for any basic set S at scale ℓ:*

- *there is a $O(\ell \cdot 2^\ell)$ time algorithm FFT_S, that when given $[f]_{\kappa^{\langle \ell \rangle}}$ as input, computes $\langle f \wr S \rangle$.*
- *there is a $O(\ell \cdot 2^\ell)$ time algorithm IFFT_S, that when given $\langle f \wr S \rangle$ as input for some $f \in \mathcal{K}^{\langle \ell \rangle}$, computes $[f]_{\kappa^{\langle \ell \rangle}}$. (In particular, $f \in \mathcal{K}^{\langle \ell \rangle}$ is uniquely specified by $\langle f \wr S \rangle$).*

Theorem 3.5 (FFT and IFFT- Univariate Polynomial Version)**.** *For each ℓ, there is a basis $\mu^{\langle \ell \rangle} = (\mu_j^{\langle \ell \rangle})_{j=0}^{2^\ell-1}$ of $\mathcal{M}^{\langle \ell \rangle}$ such that for any basic subset T of $\mathbb{F}_q$ at scale ℓ:*

- *there is a $O(\ell \cdot 2^\ell)$ time algorithm FFT_T, that when given $[g]_{\mu^{\langle \ell \rangle}}$ as input, computes $\langle g \wr T \rangle$.*
- *there is a $O(\ell \cdot 2^\ell)$ time algorithm IFFT_T, that when given $\langle g \wr T \rangle$ as input for some $g \in \mathcal{M}^{\langle \ell \rangle}$, computes $[g]_{\mu^{\langle \ell \rangle}}$. (In particular, $g \in \mathcal{M}^{\langle \ell \rangle}$ is uniquely specified by $\langle g \wr T \rangle$).*

3.1.4 FRI

Our key tool is the FRI protocol for testing proximity to univariate polynomials. Specifically, when the set of evaluation points $\mathbf{T}$ is an *evaluation domain* in $\mathbb{F}_q$, then the FFT infrastructure enables a version of the FRI protocol for $\mathsf{RS}[\mathbb{F}_q, \mathbf{T}, \rho]$, stated below. The proof appears in Appendix B.

Theorem 3.6 (Basic FRI)**.** *Let $q, \mathsf{k}, \mathsf{E}$ and the setup be as above. Let $\ell \leq \mathsf{k}$. Let $\mathcal{R}$ be a positive integer, and set $\rho = 2^{-\mathcal{R}}$. Let $\mathbf{T} \subseteq \mathbb{F}_q$ be an evaluation domain at scale ℓ with $|\mathbf{T}| = \frac{1}{\rho} 2^\ell$.*

Given a repetition parameter $t > 0$, there is an IOPP protocol (FRI) with prover P and verifier V for $\mathsf{RS}[\mathbb{F}_q, \mathbf{T}, \rho]$ with:

- **Completeness:** *There exists a prover P such that for any $f \in \mathsf{RS}[\mathbb{F}_q, \mathbf{T}, \rho]$ causes the verifier V to accept f with probability 1.*
- **Soundness:** *If f is δ far from $\mathsf{RS}[\mathbb{F}_q, \mathbf{T}, \rho]$ then for any prover P^*, we have*

$$\Pr\left[\langle \mathsf{V}(f) \leftrightarrow \mathsf{P}^*(f) \rangle = \mathsf{accept}\right] \leq \left(1 - \min\left\{\Delta(f, \mathsf{RS}[\mathbb{F}_q, \mathbf{T}, \rho]), \sqrt{\rho}\right\} + o(1)\right)^t$$

- **Prover runtime:** $O(|\mathbf{T}|)$ *arithmetic operations over $\mathbb{F}_q$*
- **Verifier runtime:** $O(t \log |\mathbf{T}|)$ *arithmetic operations over $\mathbb{F}_q$*
- **Proof length:** $O(|\mathbf{T}|)$ *field elements in $\mathbb{F}_q$.*

From the proximity gap property of Reed–Solomon codes [BCI+20], this leads to a protocol for simultaneously checking a batch of functions evaluated on an evaluation domain in $\mathbb{F}_q$ are low-degree. The proof appears in Appendix B.

Theorem 3.7 (Batched FRI)**.** *Let $q, \mathsf{k}, \mathsf{E}$ and the setup be as above. Let $\ell \leq \mathsf{k}$. Let $\mathcal{R}$ be a positive integer, and set $\rho = 2^{-\mathcal{R}}$. Let $\mathbf{T} \subseteq \mathbb{F}_q$ be an evaluation domain at scale ℓ with $|\mathbf{T}| = \frac{1}{\rho} 2^\ell$.*

Let $d_1, \ldots, d_k$ be integers such that $d_i \leq \rho |\mathbf{T}|$ for all i. Given a repetition parameter $t > 0$ and oracle access to functions

$$g_1, g_2, \ldots, g_k : \mathbf{T} \to \mathbb{F}_q,$$

there is an IOP protocol with the following behavior.

- **Completeness:** *If for all i, g_i is the evaluation of some polynomial in $\mathbb{F}_q[X]$ of degree $< d_i$, then there is a prover strategy to make the verifier accept with probability 1.*

- **Soundness:** *Suppose the protocol accepts with probability*

$$p \geq (\rho^{1/2} + \epsilon)^t + O\left(\frac{\rho^2 |\mathbf{T}|^2}{\epsilon^7 q}\right).$$

Then there exist polynomials $G_1(X), \ldots, G_k(X) \in \mathbb{F}_q[X]$, with $\deg(G_i) < d_i$ and a set $V \subseteq \mathbf{T}$ such that:

1. $|V| \geq (\rho^{1/2} + \epsilon)|\mathbf{T}|$,
2. $g_i(x) = G_i(x)$ *for all* $x \in V$, $i \in [k]$.

- **Prover runtime:** $O(k|\mathbf{T}|)$ *arithmetic operations over* $\mathbb{F}_q$
- **Verifier runtime:** $O(t(k + \log|\mathbf{T}|))$ *arithmetic operations over* $\mathbb{F}_q$
- **Proof length:** $O(|\mathbf{T}|)$ *field elements in* $\mathbb{F}_q$.

Note: *The constants in $O(\cdot)$ in the last three items in both Theorems 3.6 and 3.7 are some explicit small constants that are each at most* 10.

3.1.5 Vanishing detection

The final tool that we need is a way to check that some given rational function on E vanishes at a given set of points. This is essentially the content of Lemma 7.2 and Theorem 7.3, and is proved in Section 7.

Theorem 3.8 (Vanishing detection)**.** *Let $I \subseteq \mathsf{E}(\mathbb{F}_q)$ be a subset which is contained in a coset of $G^{\langle \ell \rangle}$. There is a well-defined rational function $\omega_I^{\langle \ell \rangle} \in \mathscr{L}([G^{\langle \ell+1 \rangle} \setminus G^{\langle \ell \rangle}] - [G^{\langle \ell \rangle}] + [I])$ on E with the following properties:*

- *For every $f \in \mathscr{L}(2[G^{\langle \ell \rangle}])$, we have:*

$$f \text{ vanishes on } I \Leftrightarrow \omega_I^{\langle \ell \rangle} \cdot f \in \mathscr{L}([G^{\langle \ell+1 \rangle}]).$$

- *For almost every $P \in \mathsf{E}(\mathbb{F}_q)$, excluding at most three cosets of $G^{\langle \ell+1 \rangle}$, $\omega_I^{\langle \ell \rangle}(P)$ can be computed using $O(\|I\| + \ell)$ $\mathbb{F}_q$-operations (where $\|I\|$ is the coset complexity of I).*

3.2 The IOP Protocol

In this section we describe an IOP for the satisfiable AIR language of Definition 2.6.

The crux of this protocol is for the prover to do a *"low-degree extension"* of a satisfying AIR-witness $\vec{\mathsf{f}} = (\mathsf{f}_1, \ldots, \mathsf{f}_\mathsf{w})$, where each $\mathsf{f}_l : \mathsf{H}_0 \to \mathbb{F}_q$. This is not the standard univariate polynomial low-degree extension; instead it is an elliptic curve variant. Indeed, we first identify H_0 with a coset C of a cyclic subgroup of size 2^h of a suitable elliptic curve E over $\mathbb{F}_q$. Thus we may view each f_l as a function defined at some points of E. Next, we consider the Riemann–Roch space $\mathcal{K}^{\langle \mathsf{h} \rangle}$ of E, and the prover finds elements $\widehat{f_l}$ of $\mathcal{K}^{\langle \mathsf{h} \rangle}$ whose restrictions to C agree with the values taken on H_0 by the f_l's. Finally, the prover provides evaluations of these rational functions $\widehat{f_l}$'s at another set of points $\mathsf{D} \subseteq \mathsf{E}(\mathbb{F}_q)$. These extended evaluations are at the core of the prover's proof of satisfiability of an AIR.

To describe the IOP for $\mathsf{L}_{\mathsf{AIR}}$ we need to fix some auxiliary parameters aux that will be used by it. For simplicity and ease of exposition, we will only describe the IOP for AIRs which have the constraint degree $\mathsf{d} = 2$.

- The *rate parameter* $\rho = 2^{-\mathcal{R}}$ for some integer $\mathcal{R}$. In practical settings, ρ is typically fixed to a small constant such as $\frac{1}{16}$ (thus $\mathcal{R} = 4$), and it may help the reader to consider this setting on first reading.

I AM IN THE MIDDLE OF CHANGING ρ from $1/16$ to general ρ.

- An elliptic curve E over $\mathbb{F}_q$ with a cyclic subgroup G of size 2^{k}, for $\mathsf{k} = \mathsf{h} + \mathcal{R} + 5$. We then use the setup from Setup 3.3 with respect to this curve.
- A choice of a coset C of $G^{\langle \mathsf{h} \rangle}$ such that $C \neq -C$. We identify H_0 with C by first picking an arbitrary $Q_0 \in C$, an arbitrary generator g of $G^{\langle \mathsf{h} \rangle}$, and identifying

  $$\mathsf{g}^j \leftrightarrow Q_0 + j \cdot g.$$

  With this identification, the constraint enforcement domains $\mathsf{H}_\alpha \subseteq \mathsf{H}_0$ get identified with $\mathsf{U}_\alpha \subseteq C$ using:

  $$\mathsf{U}_\alpha = \{Q_0 + j \cdot g \mid \mathsf{g}^j \in \mathsf{H}_\alpha\}.$$

  Note that $C \cup (-C)$ is a basic set at scale h.
- An evaluation domain $\mathbf{S} \subseteq \mathsf{E}(\mathbb{F}_q)$ at scale h (as in Section 5.3), of size $2^{\mathsf{k}'} = \mathsf{d} \cdot \frac{1}{\rho} \cdot 2^{\mathsf{h}+1} = 2^{\mathsf{h}+\mathcal{R}+1}$, which is disjoint from the trace domain H_0. Thus $\mathbf{S}$ is the union of $\frac{\mathsf{d}}{\rho} = 2^{\mathcal{R}+1}$ basic sets at scale h.
- The projection $\mathbf{T} \subseteq \mathbb{F}_q$ of the evaluation domain $\mathbf{S}$ to the x-coordinate (recall the curve is in Weierstrass form) — this is an evaluation domain of $\mathbb{F}_q$ at scale h. Note that $|\mathbf{T}| = \frac{1}{\rho} 2^{\mathsf{h}+1} = 2^{\mathsf{h}+\mathcal{R}+1}$.

  Later in the protocol, we shall represent functions $f(x, y) : \mathbf{S} \to \mathbb{F}_q$ as a pair $f_0(x), f_1(x) : \mathbf{T} \to \mathbb{F}_q$ where $\mathbf{T}$ is the projection of $\mathbf{S}$ onto the x-coordinate, using the decomposition of (4), i.e., defining

  $$f(x, y) := \frac{1}{\Omega^{\langle \ell \rangle}(x)} \left(f_0(x) + \frac{y}{x} \cdot f_1(x) \right),$$

  where f is (or is supposed to be) an evaluation of a function in $\mathcal{K}^{\langle \ell \rangle}$.

We shall also use the following notation:

- For $f : \mathbf{T} \to \mathbb{F}_q$ and a function $u : A \to \mathbb{F}_q$, where $A \cap \mathbf{T} = \emptyset$, we define the *quotient* of f by u to be the function:

  $$\mathsf{Quotient}\,(f; u) : \mathbf{T} \to \mathbb{F}_q, \quad \mathsf{Quotient}\,(f; u)\,(x) := \frac{f(x) - U(x)}{Z_A(x)},$$

  where:
  - $U(X) \in \mathbb{F}_q[X]$ is the unique polynomial of degree at most $|A| - 1$ with $U|_A = u$,
  - $Z_A(X) = \prod_{a \in A}(X - a)$ is the vanishing polynomial of A.

Description of the protocol The protocol starts with an AIR instance $\mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{I}, \mathsf{Cset})$ and auxiliary IOP parameters $\mathsf{aux} = (\mathsf{E}, G, C, \mathbf{S}, \mathsf{k}', t)$ given to both prover and verifier.

At the high level, the steps closely track the corresponding steps in the STARK protocol given in [Sta21][10], with rational functions and points on the curve replacing univariate polynomials and points in $\mathbb{F}_q$.

At some points, we represent rational functions on the elliptic curve by pairs of univariate polynomials, and invoke results about univariate polynomials. A more natural and clean version could have been given if we had analogues of (i) the proximity gaps phenomenon [BCI+20], and (ii) the DEEP query and quotienting method [BGKS20], for AG codes on elliptic curves. We believe that this approach ought to work but have not pursued these here in the interest of the simplicity of relying on previous results for RS codes.

We now give the description of the IOP protocol.

[10] Some optimizations from [Sta21], which are important for practical considerations and could also be done here, are omitted for clarity.

1. **Execution trace oracle:** The prover first finds an AIR witness $\vec{\mathsf{f}} = (\mathsf{f}_1, \ldots, \mathsf{f}_\mathsf{w})$ that satisfies the AIR instance A according to Definition 2.5. Next, the prover finds functions $\widehat{f}_1, \ldots, \widehat{f}_\mathsf{w} \in \mathcal{K}^{\langle \mathsf{h} \rangle}$ extending the f_l's. Specifically, $\widehat{f}_l$ is rational function $\widehat{f}_l(X, Y) \in \mathcal{K}^{\langle \mathsf{h} \rangle}$ such that $\widehat{f}_l\Big|_C = \mathsf{f}_l|_{\mathsf{H}_0}$.

   Note that a function $\widehat{f}_l \in \mathcal{K}^{\langle \mathsf{h} \rangle}$ can be specified by giving its values on the entire basic set $C \cup (-C)$ (using the IFFT from Theorem 3.4); thus the prover has many valid choices for $\widehat{f}_l$, determined by the values of $\widehat{f}_l\Big|_{-C}$.

   The prover then expresses each $\widehat{f}_l(X, Y)$ using a pair of univariate polynomials $\widehat{f}_{l,0}(X), \widehat{f}_{l,1}(X) \in \mathbb{F}_q[X]$ of degree $< 2^\mathsf{h}$, via the decomposition of (4), i.e.,

$$\widehat{f}_l(X, Y) := \frac{1}{\Omega^{\langle \mathsf{h} \rangle}(X)} \left(\widehat{f}_{l,0}(X) + \frac{Y}{X} \widehat{f}_{l,1}(X) \right).$$

   The prover then evaluates these $2\mathsf{w}$ low-degree polynomials $\langle \widehat{f}_{l,0}, \widehat{f}_{l,1} \mid l \in [\mathsf{w}] \rangle$ at all the points of $\mathbf{T}$.

   > Prover sends $\left\langle \widehat{f}_{l,m} \wr \mathbf{T} \right\rangle$ for each $(l, m) \in [\mathsf{w}] \times \{0, 1\}$.

   Note that these are evaluations of degree 2^h polynomials on a set $\mathbf{T}$ of size $\frac{1}{\rho} 2^{\mathsf{h}+1}$, so they are all supposed to be codewords of $\mathsf{RS}(\mathbb{F}_q, \mathbf{T}, \rho)$ (and even of $\mathsf{RS}(\mathbb{F}_q, \mathbf{T}, \rho/2)$).

2. **Constraint randomness:**

   > Verifier samples uniform randomness $\vec{\mathsf{r}} := (\mathsf{r}_1, \ldots, \mathsf{r}_\mathsf{s}) \in \mathbb{F}_q^\mathsf{s}$, one field element per constraint, and sends it to the prover.

   We now explain the role of this step. These random field elements will be coefficients for taking a "random linear combination of the constraints" – and the prover will now try to convince the verifier that this random linear combination of the constraints is satisfied by the witness underlying the $\widehat{f}_{l,0}$'s and the $\widehat{f}_{l,1}$'s.

   In more detail, constraint C_α asks that

$$\mathsf{Q}_\alpha((\mathsf{f}_l(\mathsf{g}^j \cdot t))_{l,j}) = 0,$$

   for all $t \in \mathsf{H}_\alpha$.

   If the $\widehat{f}_l \in \mathcal{K}^{\langle \mathsf{h} \rangle}$ are truly such that $\widehat{f}_l|_{\mathsf{H}_0} = \mathsf{f}_l|_C$, then this is the same as:

$$\mathsf{Q}_\alpha((\widehat{f}_l(P + j \cdot g))_{(l,j) \in \mathsf{I}}) = 0,$$

   for all $P \in \mathsf{U}_\alpha \subset \mathsf{E}$.

   Since $\widehat{f}_l \in \mathcal{K}^{\langle \mathsf{h} \rangle} = \mathscr{L}([G^{\langle \mathsf{h}+1 \rangle}])$ and Q_α has degree at most $\mathsf{d} = 2$, we get that the function $B_\alpha : \mathsf{E} \to \mathbb{F}_q$ defined by:

$$B_\alpha(P) := \mathsf{Q}_\alpha((\widehat{f}_l(P + j \cdot g))_{(l,j) \in \mathsf{I}}) \quad \forall P \in \mathsf{E},$$

   lies in $\mathscr{L}(2[G^{\langle \mathsf{h}+1 \rangle}])$. Note that the verifier can simulate oracle access to B_α at points in $\mathbf{S}$ using oracle access to evaluations of $\widehat{f}_l$ at points in $\mathbf{S}$, which themselves can be reconstituted from evaluations of $\widehat{f}_{l,0}$ and $\widehat{f}_{l,1}$ at points in $\mathbf{T}$.

   By Lemma 7.2, checking that B_α vanishes at all points in H_α is equivalent to checking that the rational function

$$\omega_\alpha \cdot B_\alpha$$

   lies in $\mathscr{L}([G^{\langle \mathsf{h}+2 \rangle}] = \mathcal{K}^{\langle \mathsf{h}+1 \rangle}$, where $\omega_\alpha := \omega_{\mathsf{H}_\alpha}$ is the degree adjustment function for U_α.

Now we can explain where the randomness $\vec{\mathsf{r}}$ is used — it is to check all the above memberships of $\omega_\alpha \cdot B_\alpha$ in $\mathcal{K}^{\langle \mathsf{h}+1\rangle}$ simultaneously. The prover will try to convince the verifier that the random linear combination:

$$\widehat{f^{\vec{\mathsf{r}}}} = \sum_\alpha \mathsf{r}_\alpha \omega_\alpha B_\alpha \tag{5}$$

lies in $\mathcal{K}^{\langle \mathsf{h}+1\rangle}$. This is what the prover does next.

3. **Constraint trace oracle:**

   The Prover then represents the rational function $\widehat{f^{\vec{\mathsf{r}}}} \in \mathcal{K}^{\langle \mathsf{h}+1\rangle}$ as 2 univariate polynomials:

$$\widehat{f^{\vec{\mathsf{r}}}}(X,Y) = \frac{1}{\Omega^{\langle \mathsf{h}+1\rangle}(X)}\left(\widehat{f_0^{\vec{\mathsf{r}}}}(X) + \frac{Y}{X}\widehat{f_1^{\vec{\mathsf{r}}}}(X)\right),$$

   where $\widehat{f_m^{\vec{\mathsf{r}}}} \in \mathcal{M}_{\mathsf{h}+1}$ for $m \in \{0,1\}$.

   The prover then evaluates both univariate polynomials at the points of $\mathbf{T}$.

   > Prover sends $\langle \widehat{f_0^{\vec{\mathsf{r}}}} \wr \mathbf{T}\rangle, \langle \widehat{f_1^{\vec{\mathsf{r}}}} \wr \mathbf{T}\rangle$.

   Note that these are evaluations of univariate polynomials of degree $< 2^{\mathsf{h}+1}$ at $2^{\mathsf{h}+\mathcal{R}+1}$ points.

4. **DEEP query:**

   > Verifier samples DEEP query $\mathsf{q} = (\mathsf{x}_0, \mathsf{y}_0)$ uniformly at random from $\mathsf{E}(\mathbb{F}_q) \setminus (\overline{C} \cup \mathbf{S})$, where $\overline{C} = G^{\langle \mathsf{h}+2\rangle} \cup (G^{\langle \mathsf{h}+2\rangle} + C) \cup (G^{\langle \mathsf{h}+2\rangle} - C)$ is a union of three cosets of $G^{\langle \mathsf{h}+2\rangle}$.

5. **DEEP answer:**

   > Prover sends an answer sequence
   >
   > $$\mathsf{answer} = \langle \langle \alpha_{j,l,0}, \alpha_{j,l,1} : (j,l) \in \mathsf{I}\rangle, \langle \beta_0, \beta_1\rangle\rangle \in \mathbb{F}_q^{\mathsf{I}\times\{0,1\}} \times \mathbb{F}_q^2.$$

   The $\alpha_{j,l,m}$ are supposed to be the evaluations $\widehat{f}_{l,m}(\mathsf{q} + jg)$, and β_m is supposed to be the evaluation $\widehat{f_m^{\vec{\mathsf{r}}}}(\mathsf{q})$. Following the DEEP philosophy [BGKS20], we can then incorporate these claimed evaluations of $\widehat{f}_{l,m}$ and $\widehat{f_m^{\vec{\mathsf{r}}}}$ by *quotienting*. This will be taken into account in the next step of the protocol.

   But first, the verifier has to do a basic sanity check on the claimed evaluations. Letting

$$\begin{aligned}
\alpha_{j,l} &:= \frac{1}{\Omega^{\langle \mathsf{h}\rangle}(\pi(\mathsf{q} + j\cdot g))}\left(\alpha_{j,l,0} + \zeta(\mathsf{q} + j\cdot g)\cdot \alpha_{j,l,1}\right) \\
\beta &:= \frac{1}{\Omega^{\langle \mathsf{h}+1\rangle}(\pi(\mathsf{q}))}(\beta_0 + \zeta(\mathsf{q})\beta_1)
\end{aligned}$$

   then supposedly $\alpha_{j,l} = \widehat{f}_l(\mathsf{q} + j\cdot g)$ and $\beta = \widehat{f^{\vec{\mathsf{r}}}}(\mathsf{q})$.

   We say the constraints Q_α are *validated* by answer if the following equality holds:

$$\sum_\alpha \mathsf{r}_\alpha \omega_\alpha(\mathsf{q}) \mathsf{Q}_\alpha\big((\alpha_{j,l})_{(j,l)\in\mathsf{I}}\big) = \beta, \tag{6}$$

   i.e., the answers are consistent with Eq. (5).

6. **FRI Protocol:** This step verifies the low-degreeness of various functions simultaneously. But first, we quotient out the functions $\widehat{f}_{l,m}$ and $\widehat{f^{\mathsf{r}}_m}$ by their evaluations that the prover claimed in the previous step.

   For $l \in [\mathsf{w}]$, define $A_l \subseteq \mathbb{F}_q$ to be the set:

   $$A_l = \{\pi(\mathsf{q} + j \cdot g) \mid (j, l) \in \mathsf{I}\}$$

   Define $u_{l,m} : A_l \to \mathbb{F}_q$ to be:

   $$u_{l,m}(\pi(\mathsf{q} + j \cdot g)) = \alpha_{j,l,m}.$$

   For $l \in [\mathsf{w}]$ and $m \in \{0, 1\}$, define $\widehat{b}_{l,m} : \mathbf{T} \to \mathbb{F}_q$ by:

   $$\widehat{b}_{l,m}(x) = \mathsf{Quotient}\left(\widehat{f}_{l,m}; u_{l,m}\right)(x),$$

   and degree parameter $d_{l,m} = 2^{\mathsf{h}} - 1 - |A_l|$.

   For $m \in \{0, 1\}$, define $u_m : \{\pi(\mathsf{q})\} \to \mathbb{F}_q$ by

   $$u_m(\pi(\mathsf{q})) = \beta_m.$$

   Now define $\widehat{b^{\mathsf{r}}_m} : \mathbf{T} \to \mathbb{F}_q$ by:

   $$\widehat{b^{\mathsf{r}}_m}(x) = \mathsf{Quotient}\left(\widehat{f^{\mathsf{r}}_m}; u_m\right)(x),$$

   and degree parameter $d_m = 2^{\mathsf{h}+1} - 2$.

   Note that oracle access to these functions can be simulated by the verifier from oracle access to $\widehat{f}_{l,m}$ and $\widehat{f^{\mathsf{r}}_m}$ on $\mathbf{T}$.

   > Prover and Verifier now run the Batched FRI protocol from Theorem 3.7 on all the $\widehat{b}_{l,m}$ and the $\widehat{b^{\mathsf{r}}_m}$ with degree parameters $d_{l,m}$ and d_m, and repetition parameter t.

   Observe that all the degree parameters are smaller than $\rho|\mathbf{T}|$ – thus the soundness of this step is governed by ρ and t.

7. **Decision:**

   > Verifier accepts iff (i) the constraints Q_α are validated by answer (i.e., equation (6) holds), and (ii) the FRI protocol accepts.

3.3 Proof of Theorem 2.10

We now prove Theorem 2.10. As usual, the most intricate part is the claim of soundness, or, in our case, knowledge soundness.

Theorem 3.9. *Let $n = 2^{\mathsf{h}} \cdot w$. Let $\eta > 0$ be an arbitrary real number. The protocol described in Section 3.2 has the following properties:*

- **Proving Complexity:** $O(2^{\mathsf{h}} \cdot \mathsf{h} \cdot w \cdot \frac{1}{\rho} + t)$,
- **Query Complexity:** $O(\mathsf{h} \cdot w \cdot t)$,
- **Verification Time:** $O(t \cdot \mathsf{h})$,
- **Proof Length:** $\frac{1}{\rho} \cdot 2^{\mathsf{h}} \cdot 2 \cdot (2w + 3)$, *which is at most* $10 \cdot \frac{1}{\rho} \cdot n$.

- **Completeness:** *If the AIR is satisifable, there is a strategy that makes the verifier accept with probability* 1.
- **Soundness and knowledge soundness:** *If some prover* P^* *can make the verifier specified above accept with probability greater than soundness error* p_0*, where*

$$p_0 = \left(\rho^{1/2} + \eta\right)^t + O\left(\frac{2^{2\mathsf{h}}}{\eta^7 q}\right), \tag{7}$$

then the AIR is satisfiable. Furthermore, there is an extractor that runs in time $\mathsf{poly}(2^{\mathsf{h}})$ *while interacting with* P^* *and outputs w.h.p. a satisfying assignment per Definition 2.5.*

3.3.1 Resources

We first verify the claims about the running times, communication and proof lengths.

An honest prover, given access to a satisfying assignment $\vec{\mathsf{f}}$ to the AIR A, will:

- perform an IFFT and some FFTs in Step 1 to compute the $\widehat{f_l}$ and the $\langle \widehat{f_l} \wr \mathbf{S} \rangle$ (in time $O(\mathsf{w} \cdot \mathsf{h} \cdot 2^{\mathsf{h}})$),
- compute $\langle B_\alpha \wr \mathbf{S} \rangle$ and ω_α for each α (in time $O\left(2^{\mathsf{h}} \cdot \sum_\alpha (\|\mathsf{Q}_\alpha\| + \|\mathsf{H}_\alpha\|)\right)$),
- Compute $\widehat{f^{\mathsf{r}}}$ and $\langle \widehat{f^{\mathsf{r}}} \wr \mathbf{S} \rangle$,
- Find $\widehat{f_l}(\mathsf{q} + j \cdot g)$ for each $(j, l) \in \mathsf{I}$, and then use this to find $\widehat{f^{\mathsf{r}}}(\mathsf{q})$.

The claimed running time of the prover thus follows easily.

The verifier running time is similarly easily seen.

The total proof length comes from $2w + 2$ functions from $\mathbf{T}$ to $\mathbb{F}_q$, along with what the prover sends during the FRI protocol: this gives the desired claim about the total proof length.

3.3.2 Completeness

The intended response of the prover is specified in the description of the IOP. The completeness is immediate from this description.

3.3.3 Soundness and knowledge soundness

Suppose the Prover has a strategy that makes the Verifier accept with probability at least p_0 as defined in Eq. (7). We will show that the AIR instance A is satisfiable, and that a knowledge extractor can find the satisfying AIR assignment. This establishes the claimed soundness of the protocol.

For knowledge soundness, one shows that this satisfying assignment can be found by an efficient knowledge extractor – given our soundness analysis, the details are almost identical to the FFT-friendly case [Sta21], and we omit them. A key role is played by the Guruswami–Sudan list decoding algorithm for Reed–Solomon codes [GS99].

Since the Prover communicates first, we may as well assume that the Prover's first message (the $\langle \widehat{f_l} \wr \mathbf{T} \rangle$) is fixed to be the one that maximizes the probability of acceptance, and thus to one which makes it at least p_0.

Consider the list of all tuples of polynomials that have high agreement with the $\langle \widehat{f_l} \wr \mathbf{T} \rangle$:

$$\mathcal{L} = \Big\{ (P_{l,m})_{l \in [\mathsf{w}], m \in \{0,1\}} \in (\mathbb{F}_q[X])^{2\mathsf{w}} \mid \deg(P_{l,m}) \leq 2^{\mathsf{h}},$$

$$\Pr_{x\in\mathbf{T}}\left[\forall(l,m)\in[\mathsf{w}]\times\{0,1\}, P_{l,m}(x)=\widehat{f}_{l,m}(x)\right]\geq\rho^{1/2}+\eta\Bigg\}.$$

By the Johnson bound (Theorem A.1), we have that

$$|\mathcal{L}|\leq\frac{1}{2\eta\sqrt{\rho}}.$$

The satisfying AIR assignment will come from one of these tuples.

Similarly, let

$$\mathcal{L}'=\Bigg\{(P_m)_{m\in\{0,1\}}\in(\mathbb{F}_q[X])^2\mid\deg(P_m)\leq 2^{\mathsf{h}+1},\ \Pr_{x\in\mathbf{T}}\left[\forall m\in\{0,1\}, P_m(x)=\widehat{f}^{\mathsf{r}}_m(x)\right]\geq\rho^{1/2}+\eta\Bigg\}.$$

By the Johnson bound, $|\mathcal{L}'|\leq\frac{1}{2\eta\sqrt{\rho}}$.

Define ϵ by:

$$\epsilon:=\frac{1}{\eta^2\rho}\left(\frac{2^{\mathsf{h}+\mathcal{R}+1}}{q}\right)+\frac{1}{\eta\sqrt{\rho}}\frac{1}{q}.$$

Since $2^{2\mathsf{h}}<q$, we get that $\epsilon=o\left(\frac{2^{2\mathsf{h}}}{\eta^7 q}\right)$ as $q\to\infty$.

Let E be the event that after the first 5 steps of the protocol, both the following happen:

- the probability of the FRI protocol accepting in step 6 is at least

$$p_0-\epsilon\geq\left(\rho^{1/2}+\eta\right)^t+O\left(\frac{2^{2\mathsf{h}}}{\eta^7 q}\right)$$

  for p_0 defined in Eq. (7); and
- Eq. (6) holds.

Then $\Pr[E]\geq\epsilon$.

E implies a special relationship between $\mathcal{L}$, $\mathcal{L}'$ and q Suppose E occurs. Then by Theorem 3.7 we get that there exists a subset $V\subseteq\mathbf{T}$, with $|V|\geq\sqrt{\rho}+\eta$ and polynomials $\widehat{B}_{l,m}(X)$ of degree at most $d_{l,m}$ and $\widehat{B}_m(X)$ of degree at most d_m such that for all $x\in V$ and all l,m:

$$\widehat{B}_{l,m}(x)=\widehat{b}_{l,m}(x),$$

$$\widehat{B}_m(x)=\widehat{b}^{\mathsf{r}}_m(x).$$

Opening up the definition of quotient in the $\widehat{b}_{l,m}$ and the $\widehat{b}^{\mathsf{r}}_m$, we get that for all $x\in V$:

$$F_{l,m}(x)=\widehat{f}_{l,m}(x),$$

$$F_m(x)=\widehat{f}^{\mathsf{r}}_m(x),$$

where:

$$F_{l,m}(X)=\widehat{B}_{l,m}(X)\cdot Z_{A_l}(X)+U_l(X)$$

is a polynomial of degree at most $d_{l,m} + |A_l| = 2^{\mathsf{h}} - 1$, and

$$F_m(X) = \widehat{B}_m(X)(X - \pi(\mathsf{q})) + \beta_m$$

is a polynomial of degree at most $d_m + 1 = 2^{\mathsf{h}+1} - 1$.

Since $|V| \geq \rho^{1/2} + \eta$, we conclude that

$$(F_{l,m})_{l \in [\mathsf{w}], m \in \{0,1\}} \in \mathcal{L}.$$
$$(F_m)_{m \in \{0,1\}} \in \mathcal{L}'.$$

Observe that each $F_{l,m}$ and F_m is a polynomial whose evaluations are consistent with the claimed DEEP answers:

$$F_{l,m}(\pi(\mathsf{q} + j \cdot g)) = \alpha_{j,l,m}.$$

$$F_m(\pi(\mathsf{q})) = \beta_m.$$

Combining the above argument with Eq. (6), the situation can be summarized as follows.

- For $\vec{P} = (P_{l,m})_{(l,m) \in \mathsf{I}} \in \mathcal{L}$, define the rational functions:

$$P_l(Z) := \frac{1}{\Omega^{\langle \mathsf{h} \rangle}(\pi(Z))} \left(P_{l,0}(\pi(Z)) + \zeta(Z) P_{l,1}(\pi(Z)) \right), l \in \{1, \ldots, \mathsf{w}\}$$
$$\gamma_{\vec{P}}(Z) := \sum_{\alpha} \mathsf{r}_\alpha \omega_\alpha(Z) \mathsf{Q}_\alpha \left((P_l(Z + j \cdot g))_{(j,l) \in \mathsf{I}} \right),$$

  Observe that $\vec{P}(Z) \in \mathscr{L}([G^{\langle \mathsf{h}+2 \rangle}] + [C])$. Indeed:

  - For each l, P_l is in $\mathcal{K}^{\langle \mathsf{h} \rangle}$ (since $\deg(P_{l,0}), \deg(P_{l,1}) \leq 2^{\mathsf{h}} - 1$), and since $\mathcal{K}^{\langle \mathsf{h} \rangle}$ is invariant under translation of the argument by g, $P_l(Z + j \cdot g)$ is also in $\mathcal{K}^{\langle \mathsf{h} \rangle}$.
  - Applying the degree 2 polynomial Q_α to a collection of elements in $\mathcal{K}^{\langle \mathsf{h} \rangle} = \mathscr{L}([G^{\langle \mathsf{h}+1 \rangle}])$ results in a rational function in $\mathscr{L}(2[G^{\langle \mathsf{h}+1 \rangle}])$.
  - Multiplying a rational function in $\mathscr{L}(2[G^{\langle \mathsf{h}+1 \rangle}])$ with ω_α results in a function in $\mathscr{L}([G^{\langle \mathsf{h}+2 \rangle}] + [C])$.

- For $\vec{W} = (W_m)_{m \in \{0,1\}} \in \mathcal{L}'$, define the rational function:

$$\lambda_{\vec{W}}(Z) := \frac{1}{\Omega^{\langle \mathsf{h}+1 \rangle}(\pi(Z))} \left(W_0(\pi(Z)) + \zeta(Z) W_1(\pi(Z)) \right).$$

  Observe that $\lambda_{\vec{W}}(Z) \in \mathcal{K}^{\langle \mathsf{h}+1 \rangle}$ (since $\deg(W_0), \deg(W_1) \leq 2^{\mathsf{h}+1} - 1$).

- If E occurs, then there is some $\vec{P} = (P_{l,m})_{(l,m) \in \mathsf{I}} \in \mathcal{L}$ and some $\vec{W} = (W_m)_{m \in \{0,1\}} \in \mathcal{L}'$ such that the low-degree rational functions $\gamma_{\vec{P}} \in \mathscr{L}([G^{\langle \mathsf{h}+2 \rangle}] + [C])$ and $\lambda_{\vec{W}} \in \mathcal{K}^{\langle \mathsf{h}+1 \rangle} = \mathscr{L}([G^{\langle \mathsf{h}+2 \rangle}])$ have the same evaluation at the point q.

$\mathcal{L}$ and $\mathcal{L}'$ have a special relationship often We know that $\Pr[E] \geq \epsilon$, and whenever E happens, there is a strong relationship between $\mathcal{L}$, $\mathcal{L}'$ and q.

Note that $\mathcal{L}$ is determined after Step 1, and $\mathcal{L}'$ is determined by Step 3 of the protocol. q is only chosen in Step 4. We now show that by Step 3 of the protocol, $\mathcal{L}$ and $\mathcal{L}'$ must already have quite a special relationship.

Let H be the event, determined after Step 3, that there exist some $\vec{P} \in \mathcal{L}$ and some $\vec{W} \in \mathcal{L}'$ such that the rational functions $\lambda_{\vec{P}}$ and $\gamma_{\vec{W}}$ are identical.

We will show that

$$\Pr[H] \geq \epsilon - \frac{1}{4\eta^2\rho}\frac{2^{\mathsf{h}+\mathcal{R}+1}}{q} > \frac{1}{\eta\sqrt{\rho}}\frac{1}{q}. \tag{8}$$

The second inequality is by definition of ϵ, and the first follows from:

Claim 3.10.

$$\Pr[E \mid \overline{H}] \leq \frac{1}{4\eta^2\rho}\frac{2^{\mathsf{h}+\mathcal{R}+1}}{q}.$$

Proof of Claim 3.10. Condition on the state of the protocol at the conclusion of Step 3. For fixed $\vec{P} \in \mathcal{L}$, $\vec{W} \in \mathcal{L}'$, let $E_{\vec{P},\vec{W}}$ denote the event that $\gamma_{\vec{P}}$ and $\lambda_{\vec{W}}$ agree at q. By the above,

$$E \subseteq \bigcup_{\vec{P}\in\mathcal{L},\vec{W}\in\mathcal{L}'} E_{\vec{P},\vec{W}}.$$

By the union bound, we get:

$$\Pr[E \mid \overline{H}] \leq \sum_{\vec{P}\in\mathcal{L},\vec{W}\in\mathcal{L}'} \Pr[E_{\vec{P},\vec{W}} \mid \overline{H}].$$

For each $\vec{P} \in \mathcal{L}$ and $\vec{W} \in \mathcal{L}'$, we know that $\gamma_{\vec{P}}$ and $\lambda_{\vec{W}}$ are distinct, low-degree rational functions (they both lie in $\mathscr{L}([G^{\langle \mathsf{h}+2\rangle}] + [C])$, and thus the probability that they agree on the randomly chosen point q (chosen uniformly from among $|\mathsf{E}(\mathbb{F}_q)| - |\mathbf{S}| - |\overline{C}| \geq q - 2\sqrt{q} - 2^{\mathsf{h}+\mathcal{R}+3}$ points) is small. Explicitly, we get:

$$\Pr[E_{\vec{P},\vec{W}} \mid \overline{H}] \leq \frac{2^{\mathsf{h}+\mathcal{R}}}{q - 2\sqrt{q} - 2^{\mathsf{h}+\mathcal{R}+3}} \leq \frac{2^{\mathsf{h}+\mathcal{R}+1}}{q},$$

and by our bounds on $|\mathcal{L}|, |\mathcal{L}'|$, Claim 3.10 follows, and with it, so does Eq. (8). □

Extracting the satisfying AIR assignment We just saw that the event H happens with noticeable probability.

If H happens, then it means that there is some $\vec{P} \in \mathcal{L}$ such that $\gamma_{\vec{P}} \in \mathcal{K}^{\langle \mathsf{h}+1\rangle}$. Recall that $\mathcal{L}$ is determined after Step 1 (and is deterministic by our assumption that the Prover's first message is deterministic), but $\gamma_{\vec{P}}$ is only determined after the randomness $\vec{\mathsf{r}} := (\mathsf{r}_1, \ldots, \mathsf{r}_\mathsf{s}) \in \mathbb{F}_q^\mathsf{s}$ is chosen in Step 2.

Let us look into the structure of the rational function $\gamma_{\vec{P}}$ (which apriori lies in $\mathscr{L}([G^{\langle \mathsf{h}+2\rangle}] + [C])$. It is the inner product between

$$\vec{\mathsf{r}} := (\mathsf{r}_1, \ldots, \mathsf{r}_\mathsf{s}) \in \mathbb{F}_q^\mathsf{s}$$

and the vector $\Gamma_{\vec{P}} \in \mathscr{L}([G^{\langle \mathsf{h}+2\rangle}] + [C])^\mathsf{s}$ given by:

$$\Gamma_{\vec{P}} = \left(\omega_\alpha(Z)\mathsf{Q}_\alpha\left((P_l(Z + j\cdot g))_{(j,l)\in\mathsf{I}}\right)\right)_{\alpha\in\{1,\ldots,\mathsf{s}\}}.$$

We say $\vec{P}$ is *good* if all entries of $\Gamma_{\vec{P}}$ lie in $\mathcal{K}^{\langle \mathsf{h}+1\rangle}$.

If $\vec{P}$ is not good, then the probability that a random linear combination of the entries of $\Gamma_{\vec{P}}$ entries lies in $\mathcal{K}^{\langle \mathsf{h}+1\rangle}$ (which is a strict subspace of the span of the entries) is at most $\frac{1}{q}$. That is,

$$\Pr[\gamma_{\vec{P}} \in \mathcal{K}^{\langle \mathsf{h}+1\rangle}] \leq \frac{1}{q}.$$

If all the $\vec{P} \in \mathcal{L}$ are not good, we get:

$$\begin{aligned}\Pr[H] &\leq \Pr[\exists \vec{P} \in \mathcal{L} \text{ s.t. } \gamma_{\vec{P}} \in \mathcal{K}^{\langle \mathsf{h}+1 \rangle}] \\ &\leq \sum_{\vec{P} \in \mathcal{L}} \Pr[\gamma_{\vec{P}} \in \mathcal{K}^{\langle \mathsf{h}+1 \rangle}] \\ &\leq \frac{1}{2\eta\sqrt{\rho}} \cdot \frac{1}{q},\end{aligned}$$

which contradicts our lower bound (Eq. (8)) on $\Pr[H]$.

Thus some $\vec{P} \in \mathcal{L}$ is good. This will give us our satisfying AIR assignment. For each $l \in [\mathsf{w}]$, let $f_l^\star = P_l \in \mathcal{K}^{\langle \mathsf{h} \rangle}$ for some fixed good $\vec{P}$. Then $f_l^\star|_C$ gives us a function defined on H_0 (via the identification of H_0 with the coset C). We claim that this is the desired satisfying AIR assignment.

Fix α. By the goodness of $\vec{P}$, we get that the rational function:

$$\omega_\alpha(Z) \cdot Q_\alpha\left((f_l^\star(Z + j \cdot g))_{(j,l) \in \mathsf{I}}\right)$$

lies in $\mathcal{K}^{\langle \mathsf{h}+1 \rangle}$, which means (by the defining property of ω_α) that the rational function

$$B_\alpha^\star(Z) = Q_\alpha\left((f_l^\star(Z + j \cdot g))_{(j,l) \in \mathsf{I}}\right)$$

vanishes at all points is U_α (which is identified with the α-th enforcement domain H_α). This is precisely the statement that $(f_l^\star)_{l \in [\mathsf{w}]}$ satisfies the α-th constraint.

Since this holds for each α, we conclude that $f_1^\star, \ldots, f_{\mathsf{w}}^\star$ is a satisfying AIR assignment per Definition 2.5, as desired.

4 Sequence of Elliptic Curve isogenies

The backbone of all of the constructions in this paper is the chain of 2-isogenies whose existence was shown in [BCKL21, Theorem 4.9], which we quoted earlier and restate here:

Theorem 3.1 (Good Curve Sequence)**.** *For any prime power $q \geq 7$ and any $1 < \mathsf{K} = 2^{\mathsf{k}} \leq 2\sqrt{q}$, there exist elliptic curves $E_0, E_1, \ldots, E_{\mathsf{k}}$ over $\mathbb{F}_q$ in extended Weierstrass form, a subgroup $G_0 \subseteq E_0$ of size K, 2-isogenies $\varphi_i : E_i \to E_{i+1}$ and rational functions $\psi_i : \mathbb{P}^1 \to \mathbb{P}^1$ of degree 2, such that the following diagram is commutative:*

$$\begin{array}{ccccccc} E_0 & \xrightarrow{\varphi_0} & E_1 & \xrightarrow{\varphi_1} & \cdots & \xrightarrow{\varphi_{\mathsf{k}-1}} & E_{\mathsf{k}} \\ \downarrow{\scriptstyle \pi_0} & & \downarrow{\scriptstyle \pi_1} & & & & \downarrow{\scriptstyle \pi_{\mathsf{k}}} \\ \mathbb{P}^1 & \xrightarrow{\psi_0} & \mathbb{P}^1 & \xrightarrow{\psi_1} & \cdots & \xrightarrow{\psi_{\mathsf{k}-1}} & \mathbb{P}^1 \end{array} \tag{3}$$

where:

- *π_i are the projection maps to the x-coordinate of each curve;*
- *$|\varphi_{i-1} \circ \cdots \circ \varphi_0(G_0)| = \frac{1}{2^i}|G_0| = 2^{\mathsf{k}-i}$.*
- *G_0 has a coset C such that $C \neq -C$ (as elements of the quotient group $E_0(\mathbb{F}_q)/G_0$).*

Note that this theorem is very abstract: It only establishes the existence of these curves and maps, but says almost nothing about how the form of the equations defining E_i or of the isogenies φ_i and maps ψ_i, does not specify the structure of the 2-groups G_i, and does not show how to find such curves.

In this section we will restrict the form of the curves E_i and obtain explicit formulas for φ_i and ψ_i; we will restrict G_0 to be a *cyclic* group isomorphic to $\mathbb{Z}/2^{\mathsf{k}}\mathbb{Z}$ (and thus all other G_i-s are also cyclic) and show that it is a feasible requirement; and outline a probabilistic algorithm for finding such curves efficiently. More specifically, we show:

There is an efficient algorithm, $\mathsf{FindCurve}(q, \mathsf{k})$, which yields an elliptic curve E_0 over $\mathbb{F}_q$ with cyclic subgroup G_0 of order 2^{k}; we prove it runs in quasilinear time for prime fields (Theorem 4.8) and conjecture the same holds over all fields (Conjecture 4.9). This curve can then be presented as a *good curve* (Definitions 4.1 and 4.2 and Lemma 4.5), from which an isogeny chain as in Eq. (3) can be constructed (Section 4.2), where all curves $E_1, \ldots, E_{\mathsf{k}-1}$ are good curves and all isogenies $\varphi_0, \ldots, \varphi_{\mathsf{k}-1}$ are *good isogenies* (Lemmas 4.3 and 4.4).

4.1 Explicit curves and isogenies

In this subsection, we define particularly nice forms for our desired curves: one form for odd q, and one for even q. These forms will have the property that once E_0 is of this form, then all the rest of the curves in the chain are, too. They will also all share the same 2-torsion point $\mathbf{0} = (0, 0)$, and all isogenies will be such that $\ker \varphi_i = \{\mathbf{0}, \boldsymbol{\infty}\}$, leading to efficiently computable formulas for φ_i and ψ_i, of the same form at every level.

We first present all relevant definitions and statements about these forms.

Definition 4.1 (Good curve over odd size fields)**.** *Let q be an odd prime power. Denote by $E_{a,B}$ the elliptic curve over $\mathbb{F}_q$ given by the Weierstrass equation*

$$E_{a,B}: \qquad Y^2 = X^3 + aX^2 + BX.$$

Note that for the curve to be non-singular, we must have $B, a^2 - 4B \neq 0$.

We say that the curve $E_{a,B}$ is good *if $B = b^2 \neq 0$ is a non-zero quadratic residue, and $a + 2b$ is a quadratic residue. In this case the point $P = \left(b, b\sqrt{a+2b}\right)$ is an $\mathbb{F}_q$-point on the curve, which we call its* good point*.*

Definition 4.2 (Good curve over even size fields)**.** *Let q be a power of 2. Denote by E_B the elliptic curve over $\mathbb{F}_q$ given by the Weierstrass equation*

$$E_B: \qquad Y^2 + XY = X^3 + BX.$$

Note that the curve is non-singular iff $B \neq 0$, and all such curves will be called good*. Write $B = b^2$ with $b \in \mathbb{F}_q$ (always possible in characteristic 2). The* good point *of E_{b^2} is $P = (b, b)$.*

These curves have the following properties: All curves $E_{a,B}$ and E_B pass through the points $\boldsymbol{\infty} = [0 : 1 : 0]$ and $\mathbf{0} = (0, 0)$, and $\mathbf{0}$ is a 2-torsion point, as mentioned above. For good curves, the good point P is always a 4-torsion point, with $2P = \mathbf{0}$ (the conditions for being a good curve are exactly the conditions for the existence of such a point). Finally, the good curves all have nice isogenies to other curves of similar forms, as described by the following lemmas:

Lemma 4.3 (Good isogenies over odd size fields)**.** *Let $E = E_{a,b^2}$ be a good curve in odd characteristic, with good point P. Let $E' = E_{a+6b,4ab+8b^2}$. Then there is a 2-isogeny $\varphi = \varphi_b : E \to E'$ given by*

$$\varphi(x, y) = \left(x - 2b + \frac{b^2}{x}, \left(1 - \frac{b^2}{x^2}\right) y\right).$$

Furthermore, we have $\ker \varphi = \{\mathbf{0}, \boldsymbol{\infty}\}$ and $\varphi^{-1}(\mathbf{0}) = \{\pm P\}$. We call this φ the good isogeny *of E.*

Lemma 4.4 (Good isogenies over even size fields)**.** *Let $E = E_{b^2}$ be a good curve in even characteristic, with good point P. Denote $E' = E_b$. Then there is a 2-isogeny $\varphi = \varphi_b : E \to E'$ given by*

$$\varphi(x, y) = \left(x + \frac{b^2}{x}, \frac{b(b+x)}{x} + \left(1 + \frac{b^2}{x^2}\right) y\right)$$

Furthermore, we have $\ker \varphi = \{\mathbf{0}, \boldsymbol{\infty}\}$ and $\varphi^{-1}(\mathbf{0}) = \{\pm P\}$. We call this φ the good isogeny *of E.*

We also remark that these forms do not restrict us in any way; in either odd or even characteristic, any curve with a 4-torsion point can be expressed as a good curve, after an appropriate change of coordinates. Formally, we have:

Lemma 4.5. *Given an elliptic curve E over any finite field with a 4-torsion point P, it is isomorphic to a good curve with good point that corresponds to P.*

We now explore and prove the above properties and statements, dealing separately with the odd and even cases.

4.1.1 Odd characteristic case

Let $E_{a,B}$ be a curve as in Definition 4.1. We explore the conditions for there being a point $P = (b, y_b) \in E(\mathbb{F}_q)$ with $2P = \mathbf{0}$. Since $\mathbf{0} = -\mathbf{0}$, this is equivalent to the tangent to the curve at P passing through $\mathbf{0}$. Equivalently, the vector $\overrightarrow{\mathbf{0}P} = (b, y_b)$ must be parallel to the tangent at P, which is equivalent to being perpendicular to the gradient of the curve equation at P, i.e. to $\nabla_{(b,y_b)} E_{a,B}$. Thus $2P = \mathbf{0}$ is equivalent to

$$0 = \langle \nabla_{(b,y_b)} E_{a,B}, (b, y_b) \rangle = 2y_b^2 - 3b^3 - 2ab^2 - Bb = -b^3 + Bb.$$

Note that $b \neq 0$ (since otherwise it would imply $2P = \boldsymbol{\infty}$), so we can divide the identity by b to obtain $B = b^2$. In other words, such curves are of the form

$$E_{a,b^2}: \qquad Y^2 = X^3 + aX^2 + b^2 X,$$

with the point P being $\left(b, b\sqrt{a+2b}\right)$, as per the definition of a good curve and point. Note that in general there may be multiple viable options for P: Clearly $-P = \left(b, -b\sqrt{a+2b}\right)$ always works, and when the 2-torsion is of size 4 (which is equivalent to $a^2 - 4b^2$ and $a - 2b$ also being quadratic residues) then the points $P' = \left(-b, \pm b\sqrt{a-2b}\right)$ also satisfy $2P' = \mathbf{0}$. Our definition partially eliminates this ambiguity: the statement "E_{a,b^2} is a good curve with good point P" always means $P_x = b$, and never allows for $P_x = -b$. In other words, writing $B = b^2$ explicitly requires a specific choice of square-root b of B, and this choice will always be equal to the x-coordinate of the good point P.

We now prove the properties of the good isogeny, described in Lemma 4.3:

Proof of Lemma 4.3. We first demonstrate this is a curve morphism, i.e. that if $(x, y) \in E$, then $(x', y') = \varphi(x, y) \in E'$. From the definition of φ we have

$$y' = \left(1 - \frac{b^2}{x^2}\right) y, \quad x' = \left(1 - \frac{b}{x}\right)^2 x$$

Thus

$$\begin{aligned} \frac{y'^2}{x'} &= \frac{1}{x'}\left(1 - \frac{b^2}{x^2}\right)^2 y^2 = \frac{1}{x}\left(1 + \frac{b}{x}\right)^2 (x^3 + ax^2 + b^2 x) = \frac{(x+b)^2(x^2+ax+b^2)}{x^2} \\ &= \frac{1}{x^2}\left(x^4 + (a+2b)x^3 + (2ab+2b^2)x^2 + (ab^2+2b^3)x + b^4\right), \end{aligned}$$

and on the other hand

$$\begin{aligned} x'^2 + (a+6b)x' + (4ab+8b^2) &= \frac{(x-b)^4 + (a+6b)(x-b)^2 x + (4ab+8b^2)}{x^2} \\ = \frac{1}{x^2}\big((x^4 - 4bx^3 + 6b^2x^2 - 4b^3x + b^4) &\\ + ((a+6b)x^3 - (2ab+12b^2)x^2 + (ab^2+6b^3)x) + (4ab+8b^2)x^2\big) &\\ = \frac{1}{x^2}\left(x^4 + (a+2b)x^3 + (2ab+2b^2)x^2 + (ab^2+2b^3)x + b^4\right) &= \frac{y'^2}{x'}, \end{aligned}$$

as claimed. Note that φ is not immediately defined at ∞ and $\mathbf{0}$ (even in projective coordinates) but using the orders of zeros/poles of X and Y at $\mathbf{0}$ and ∞ it is easy to see that $\varphi(\mathbf{0}) = \varphi(\infty) = \infty$, and in particular φ is an isogeny. It is also clear that $\varphi(Q) \neq \infty$ for any $Q \notin \{\mathbf{0}, \infty\}$ (since $Q_x \neq 0$), thus $\ker\varphi = \{\mathbf{0}, \infty\}$ and φ is a 2-isogeny. Finally,

$$\varphi(Q) = \mathbf{0} \Leftrightarrow \varphi(Q)_x = 0 \Leftrightarrow \left(1 - \frac{b}{Q_x}\right)^2 = 0 \Leftrightarrow Q_x = b \Leftrightarrow Q = \pm P,$$

thus $\varphi^{-1}(\mathbf{0}) = \pm P$. □

Remark 4.6. Since $(a+6b)^2 - 4(4ab+8b^2) = a^2 - 4ab + 4b^2 = (a-2b)^2$ is always a quadratic residue, $E'(\mathbb{F}_q)[2]$ is always of size 4, containing the points $\{\infty, \mathbf{0}, (-4b, 0), (-a-2b, 0)\}$. If $\#E(\mathbb{F}_q)[2] = 4$ as well, then the other good point candidates satisfy $\varphi((-b, \pm b\sqrt{a-2b})) = (-4b, 0)$ and the other 2-torsion points satisfy $\varphi((\frac{-a \pm \sqrt{a^2-4b^2}}{2}, 0)) = (-a-2b, 0)$. Otherwise, $(-4b, 0), (-a-2b, 0)$ are not in the image of φ. We do not use these properties in the rest of the paper.

4.1.2 Even characteristic case

Let E_B be a curve as in Definition 4.2. To find conditions for existence of $P \in E(\mathbb{F}_q)$ with $2P = \mathbf{0}$, let us calculate the x-coordinate of the formula for doubling a point on E_B. Differentiate the equation to compute the tangent at (x_0, y_0):

$$\lambda = \left(-\frac{\partial E_B}{\partial X} \Big/ \frac{\partial E_B}{\partial Y}\right)\Bigg|_{(x_0,y_0)} = \frac{y_0 + x_0^2 + B}{x_0}.$$

The tangent equation is then $Y = \lambda X + \mu$ for some μ. Substitute $\lambda X + \mu$ for Y in the curve equation to get a polynomial equation in X, with a double root at x_0 and another root at $x_1 = (2(x_0, y_0))_x = (-2(x_0, y_0))_x$:

$$(\lambda X + \mu)^2 + X(\lambda X + \mu) + X^3 + BX = 0.$$

This is a monic polynomial, thus the coefficient of X^2 equals the sum of its roots, i.e.

$$x_1 = x_0 + x_0 + x_1 = \lambda^2 + \lambda = \frac{y_0^2 + x_0^4 + B^2}{x_0^2} + \frac{y_0 + x_0^2 + B}{x_0} = \frac{x_0^4 + B^2}{x_0^2},$$

where in the last transition we used the fact that $y_0^2 + x_0 y_0 + x_0^3 + B x_0 = 0$. Thus in general:

$$(2(x, y))_x = \frac{x^4 + B^2}{x^2}.$$

This means that $\mathbf{0}$ is the only 2-torsion point on E_B other than ∞ and that the points $\pm P = \{(b, 0), (b, b)\} \subset E_B$, where $b^2 = B$, satisfy $2P = \mathbf{0}$ and are the only 4-torsion points of E_B. Again note that squaring is an automorphism of $\mathbb{F}_q$, so such b uniquely exists. Thus, as per Definition 4.2, all curves E_{b^2} are good and the good point $P = (b, b)$ satisfies $2P = \mathbf{0}$. Note that we may as well have defined $P = (b, 0)$ as the good point with no other changes.

We now prove the properties of the good isogeny, described in Lemma 4.4:

Proof of Lemma 4.4. We first show that $(x', y') = \varphi(x, y) \in E'$. Denote $r = 1 + \frac{b}{x}$, then $x' = r^2 x$ and $y' = br + r^2 y$. Thus

$$\begin{aligned} y'^2 + x'y' &= b^2 r^2 + r^4 y^2 + br^3 x + r^4 xy = r^4\left(y^2 + xy\right) + b^2 r^2 + br^3 x \\ &= r^4\left(x^3 + b^2 x\right) + b^2 r^2 + br^3 x = r^4 (x+b)^2 x + br^2 (b + rx) \\ &= r^6 x^3 + br^2 x = x'^3 + bx'. \end{aligned}$$

The kernel of φ is at the poles of the rational function from E to $\mathbb{P}^1$ given by the x-coordinate of φ, i.e. $x' = r^2 x = \frac{(x+b)^2}{x}$, which are at $\mathbf{0}$ and ∞. Likewise, $\varphi^{-1}(\mathbf{0})$ are the zeros of x', which are the points with $x = b$, namely $\{\pm P\}$. □

Remark 4.7. In the even-characteristic case, the good isogeny φ is the dual of the Frobenius isogeny $F : E' \to E$ which maps $(x, y) \in E'$ to $(x^2, y^2) \in E$: The composition $F \circ \varphi$ is the doubling endomorphism of E, and $\varphi \circ F$ is the doubling endomorphism of E'. Equivalently, one can define φ as the composition of the inverse of Frobenius on the doubling endomorphism.

4.1.3 Moving to a good curve

Finally, we prove Lemma 4.5, that every curve with a 4-torsion point can be expressed as a good curve.

Proof of Lemma 4.5. If the characteristic is odd (including 3), E is isomorphic to a curve of the form

$$Y^2 = X^3 + a_2X^2 + a_4X + a_6.$$

In that representation the y coordinate of $2P$ is 0 because $2P$ is a 2-torsion point. Applying the change of variables $X' = X - x_0, Y' = Y$, where x_0 is the x coordinate of $2P$, maps $2P$ to the point $(0, 0)$ in coordinates (X', Y'), thus in these coordinates the equation of the curve takes the form:

$$Y'^2 = X'^3 + a'_2X'^2 + a'_4X'$$

and $2P' = \mathbf{0}$, i.e. P' is a good point.

Now assume that the characteristic is 2. Then E is isomorphic to a curve of the form

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6$$

where $a_1 \neq 0$, since otherwise E is either singular (if $a_3 = 0$) or supersingular, and then it has no 4-torsion point. Substituting $X = X' + \frac{a_3}{a_1}$, the left hand side then becomes $Y^2 + a_1X'Y$ and the right hand side stays a monic polynomial in X' of degree 3. Thus our equation is

$$Y^2 + a_1X'Y = X'^3 + a'_2X'^2 + a'_4X' + a'_6.$$

We next substitute $Y = Y' + \gamma$, where $\gamma^2 = a'_6$, and get

$$Y'^2 + a_1X'Y' = X'^3 + a'_2X'^2 + a''_4X'$$

where $a''_4 = a'_4 + a_1\gamma$. In these coordinates $(x', y') = (0, 0)$ is the only 2-torsion point, as $X' = 0$ is the only vertical line that intersects E exactly once (and more generally, we have $-(x', y') = (x', y' + a_1x')$) . In particular $(2P')_{x'} = 0$, where $P' = (x'_1, y'_1)$ is the representation of P in (x', y') coordinates. Computing as in Section 4.1.2 we find the doubling formula and substitute P':

$$0 = (2P')_{x'} = \frac{x_1'^4 + a_4''^2}{a_1^2 x_1'^2},$$

hence $x_1'^2 = a''_4$. Finally, we make the last change of variables: $X' = a_1^2X''$ and $Y' = a_1^3Y'' + a_1^2\frac{y'_1}{x'_1}X''$. The curve equation becomes

$$a_1^6Y''^2 + a_1^4\frac{y_1'^2}{x_1'^2}X''^2 + a_1^6X''Y'' + a_1^5\frac{y'_1}{x'_1}X''^2 = a_1^6X''^3 + a'_2a_1^4X''^2 + a''_4a_1^2X''.$$

Since $a_1 \neq 0$ we can divide both sides by a_1^6. Additionally, we use the known curve equation on (x'_1, y'_1) together with $x_1'^2 = a''_4$ to erase the coefficient of X''^2, which is

$$a_1^4\frac{y_1'^2 + a_1x'_1y'_1 + a'_2x_1'^2}{x_1'^2} = a_1^4\frac{x_1'^3 + a''_4x'_1}{x_1'^2} = 0.$$

The final curve equation is then

$$Y''^2 + X''Y'' = X''^3 + \frac{a_4''}{a_1^4}X'',$$

which is exactly E_{b^2} for $b = \frac{x_1'}{a_1^2}$ as claimed.

□

4.2 An isogeny chain of good curves

In this subsection we show that any elliptic curve E with a large cyclic subgroup of order 2^{k} gives rise to an isogeny chain where all curves and all isogenies are good.

Suppose that $E = E_0$ is an elliptic curve over $\mathbb{F}_q$ (any finite field) such that $E(\mathbb{F}_q)$ contains a cyclic subgroup of order 2^{k} with $\mathsf{k} \geq 2$, and let $g = g_0$ be the generator of such a subgroup. According to Lemma 4.5, by applying projective transformations we may assume that E is a good curve with good point $P_0 = 2^{\mathsf{k}-2}g_0$, with $E = E_{a_0,b_0^2}$ if the characteristic is odd, or $E = E_{b_0^2}$ if the characteristic is 2. Lemmas 4.3 and 4.4 give us the good 2-isogeny $\varphi_0 = \varphi_{b_0} : E_0 \to E_1$, with $\varphi_0(\mathbf{0}) = \boldsymbol{\infty}, \varphi_0(2^{\mathsf{k}-2}g_0) = \mathbf{0}$. It follows that $g_1 = \varphi_0(g_0)$ generates a cyclic subgroup of order $2^{\mathsf{k}-1}$ inside E_1, and satisfies $2^{\mathsf{k}-2}g_1 = \mathbf{0}$. If $\mathsf{k} > 2$, we may continue similarly:

For the general iteration $0 \leq i \leq \mathsf{k}-2$, we have a curve E_i with a point g_i of order $2^{\mathsf{k}-i}$ satisfying $2^{\mathsf{k}-i-1}g_i = \mathbf{0}$. It follows that E_i (which equals E_{a_i,b_i^2} if the characteristic is odd, or $E_{b_i^2}$ if the characteristic is 2) is a good curve with good point $P_i = 2^{\mathsf{k}-i-2}g_i$ (note that P_i is well defined, and b_i is chosen to equal $(P_i)_x$). Lemmas 4.3 and 4.4 then give us the good 2-isogeny $\varphi_i : E_i \to E_{i+1}$, with $g_{i+1} = \varphi_i(g_i)$ of order $2^{\mathsf{k}-i-1}$ satisfying $2^{\mathsf{k}-i-2}g_{i+1} = \mathbf{0}$. The final curve $E_{\mathsf{k}-1}$ need not necessarily be good, and $b_{\mathsf{k}-1}$ is not necessarily defined, but $b_{\mathsf{k}-1}^2$ is still meaningful.

Considering the x-projection maps $\pi_i : E_i \to \mathbb{P}^1$, the x-coordinate of φ_i is the degree 2 rational function $\psi_i : \mathbb{P}^1(\mathbb{F}_q) \to \mathbb{P}^1(\mathbb{F}_q)$ given by

$$\psi_i(x) := \begin{cases} \frac{(x-b_i)^2}{x} & x \notin \{0,\infty\} \\ \infty & x \in \{0,\infty\} \end{cases}$$

and the following diagram commutes:

$$\begin{array}{ccccccc} E_0 & \xrightarrow{\varphi_0} & E_1 & \xrightarrow{\varphi_1} & \cdots & \xrightarrow{\varphi_{\mathsf{k}-2}} & E_{\mathsf{k}-1} \\ \downarrow{\scriptstyle \pi_0} & & \downarrow{\scriptstyle \pi_1} & & & & \downarrow{\scriptstyle \pi_{\mathsf{k}}} \\ \mathbb{P}^1 & \xrightarrow{\psi_0} & \mathbb{P}^1 & \xrightarrow{\psi_1} & \cdots & \xrightarrow{\psi^{(\mathsf{k}-2)}} & \mathbb{P}^1 \end{array} \tag{9}$$

4.3 Finding the start of a long isogeny chain

As explained in Section 4.2, an isogeny chain of length k can be constructed from an elliptic curve with a point of order 2^{k}. In [BCKL21] it was shown that for a finite field of size q such curve exists for any $\mathsf{k} \leq \frac{1}{4}\log_2 q$, but the question of how to find it was not investigated. This section provides a probabilistic algorithm, called FindCurve, for efficiently finding such curves, with the following proven and conjectural runtime bounds:

Theorem 4.8. *Let q be a prime and $\mathsf{k} \leq \frac{1}{2}\log_2 q - 1$. The algorithm $\mathsf{FindCurve}(q, \mathsf{k})$ will find a curve over $\mathbb{F}_q$ with a point of order 2^{k} using, on average, at most $O(2^{\mathsf{k}} \log^2 q \min(\mathsf{k}, \log\log q))$ field operations.*

Assuming the generalized Riemann Hypothesis, the bound can be replaced by $O(2^{\mathsf{k}} \log q \log\log q \min(\mathsf{k}, \log\log q))$ field operations.

Conjecture 4.9. *Let q be a prime power and $\mathsf{k} \leq \frac{1}{2}\log_2 q + 1$. The algorithm $\mathsf{FindCurve}(q, \mathsf{k})$ will find a curve over $\mathbb{F}_q$ with a point of order 2^{k} using, on average, $O(2^{\mathsf{k}} \log q)$ field operations.*

The conjectural bound is based on commonly accepted heuristics as well as experimental implementation.

The main idea of this method is to exhaustively pick random elliptic curves and compute their maximal 2-subgroups until we get one with a subgroup of order 2^k. We then use the found curve to find another elliptic curve of the same order, in which the maximal 2-subgroup is cyclic.

To bound the expected number of elliptic curves that will be picked this way we cite Proposition 4.11, proven by Lenstra in [Len87], where he deals with a similar problem of sampling curves until one is of order divisible by a given prime for the famous ECM factoring algorithm.

Proposition 4.10 (Lenstra)**.** *There exist effectively computable positive constants c_1, c_2 such that for each prime number $p > 3$ the following two assertions are valid:*

- *If S is a set of integers s with $|s - (p+1)| \leq 2\sqrt{p}$ then*

$$\#'\{E : E \text{ elliptic curve over } \mathbb{F}_p, |E(\mathbb{F}_p)| \in S\}/\cong_{\mathbb{F}_p} \leq c_1 |S| \sqrt{p} (\log p)(\log\log p)^2.$$

- *If S is a set of integers s with $|s - (p+1)| \leq \sqrt{p}$ then*

$$\#'\{E : E \text{ elliptic curve over } \mathbb{F}_p, |E(\mathbb{F}_p)| \in S\}/\cong_{\mathbb{F}_p} \geq c_2 (|S| - 2)\sqrt{p}/\log p.$$

Here $\cong_{\mathbb{F}_p}$ denotes the isomorphism over $\mathbb{F}_p$ relation and $\#'A$ of a set A of elliptic curves denotes the weighted size $\sum_{E \in A} \frac{1}{|Aut(E)|}$ of the set.

Lenstra proves this result only for prime fields, as that was the sole focus of his paper. Nonetheless, his method can be immediately applied for proving similar claims over all finite fields, with a few extra conditions. We state here only the generalization of the lower bound, using the same notation as above;

Proposition 4.11. *Let $q = p^e$ be a prime power. If S is a set of integers s with $|s-(q+1)| \leq \sqrt{q}$ and $s \neq 1 \pmod p$, then*

$$\#'\{E : E \text{ elliptic curve over } \mathbb{F}_q, |E(\mathbb{F}_q)| \in S\}/\cong_{\mathbb{F}_q} \geq c_2 (|S| - 2)\sqrt{q}/\log q.$$

Note that the constant c_2 is the same as in Lenstra's result. Lenstra comments that if the generalized Riemann hypothesis is assumed then the $\log p$ in the denominator of the lower bound can be replaced by $\log\log p$, and $|S| - 2$ may be replaced by $|S|$. The same is true in our generalized result.

Earlier in that paper, Lenstra observes that the number of (a, b) pairs of $\mathbb{F}_p$ elements such that the curve $Y^2 = X^3 + aX + b$ is isomorphic to a given elliptic curve E is equal to $\frac{p-1}{|Aut(E)|}$, and the same is true for prime powers. Together with **??** this yields the following corollary:

Corollary 4.12. *For each every prime power $q = p^e$ and natural number $v < \sqrt{q}/2$, when a and b are sampled uniformly at random from $\mathbb{F}_q$, the random curve E: $Y^2 = X^3 + aX + b$ satisfies:*

$$\Pr(E \text{ is ordinary} \wedge |E| \equiv 0 \pmod v) = \Omega\left(\frac{1}{v \log q}\right).$$

Proof. □

We use the lower bound in Corollary 4.12, with $v = 2^k$ to claim that if $q = p$ is prime and $2^k < \frac{\sqrt{q}}{2}$ then the expected number of elliptic curves that will be considered in FindCurve is $O(2^k \log q)$.

Nevertheless, for $v < 2\sqrt{q}$, all the sets of numbers in the Hasse bound that correspond to the different residue classes modulo v are of almost the same size. This is still true when considering a prime power q instead of p. Moreover, by [Deu41, Wat69], we know that any number $N = q + 1 - t$ such that $|t| \leq 2\sqrt{q}$ and t is coprime to q, has an elliptic curve E over $\mathbb{F}_q$ with $|E| = N$. These facts, together with known empirical observations, make it reasonable to believe in Conjecture 4.13, which reduces the number of elliptic curves that will be considered in FindCurve to $O(2^k)$.

Conjecture 4.13. *For each odd prime power q and a power of two $2^l \leq \sqrt{q}/2$, when a and b are sampled uniformly at random from $\mathbb{F}_q$, the random curve $E\colon\ Y^2 = X^3 + aX + b$ satisfies:*

$$\Pr\big(E \text{ is ordinary} \wedge |E| \equiv 0 \pmod{2^l}\big) = \Theta\big(2^{-l}\big).$$

For binary fields we conjecture the same bound, only we choose the curve directly in the form E_B when B is chosen uniformly from $\mathbb{F}_q^*$.

To compute the maximal 2-subgroup of E without first computing its size, we use the algorithm given in [MMCRV05], which terminates after $O(l)$ calls to a univariate quadratic equation solver, where 2^l is the maximal power of 2 that divides $|E|$. An additional cubic equation needs to be solved if the equation is not given in one of the good forms $E_{a,B}$ or E_B. Solving a univariate polynomial of degree ≤ 3 can be done in $O(\log q)$ field operations using the Cantor–Zassenhaus algorithm [**?**].

In the context of FindCurve, we consider the average running time of a randomly chosen input. Considering the algorithm of [MMCRV05] as a loop with l steps, let P_i denote the probability of entering the i-th iteration, which equals the probability of $|E|$ being divisible by 2^i. Thus each loop executes a total of $O(\sum_{i=1}^{\mathsf{k}} P_i)$ iterations in expectation. This is of course bounded by $O(\mathsf{k})$, but this bound can be improved for large k. By Corollary 4.12, if $q = p > 3$ is prime, then $P_i = O\Big(\frac{(\log p)(\log\log p)^2}{2^i}\Big)$, which improves on $P_i \leq 1$ starting from $i_0 = \log\log p + 2\log\log\log p$. Thus, using the trivial bound for the first i_0 iterations and Corollary 4.12 for the remaining iterations, we find

$$O\left(\sum_{i=1}^{\mathsf{k}} P_i\right) = O\left(\log\log p + 2\log\log\log p + \sum_{j\geq 0}\frac{1}{2^j}\right) = O(\log\log p).$$

If Conjecture 4.13 is true, then for any prime power q we have $P_i = O(2^{-i})$, hence $O(\sum_{i=1}^{\mathsf{k}} P_i) = O(1)$.

Once we find an ordinary[11] elliptic curve E whose size is divisible by 2^{k}, we use Sutherland's [**?**] FindFloor algorithm, which finds an elliptic curve E' isogenous to E and with a cyclic subgroup of order 2^{k}, in $O(\delta \log q)$ field operations. The $\log q$ term stands for the running time of finding all the 2-isogenies of a given curve and δ denotes the distance of E from V_d (the "floor") in the 2-isogenies graph.

Sutherland describes the components of this graph as "volcanoes" whose vertices are partitioned into one or more levels $V_0, \ldots, V_d$ such that the following hold:

- The subgraph on V_0 (the surface) is a regular graph of degree at most 2;
- For $i > 0$, each vertex in V_i has exactly one neighbor in level V_{i-1}, and this accounts for every edge not on the surface; and
- For $i < d$, each vertex in V_i has degree 3.

It follows that $2|V_i| = |V_{i+1}|$ for all $0 < i < d$ and $|V_0| \leq |V_1|$ if V_1 is nonempty. In particular, for $\delta \leq d$, at least $2^{-\delta-1}$ of the vertices lie on $V_{d-\delta}$. Thus, the average δ is $O(\sum_{\delta\geq 0} \delta \cdot 2^{-\delta-1}) = O(1)$. We conclude that FindFloor takes $O(\log q)$ field operations in expectation on an equidistributed random curve of given order.

The above analysis is not valid for the exceptional case of components containing the curves with j-invariants 0 and 1728. These components are not classical volcanoes, but the analysis of FindFloor for them is similar.

[11] for a supersingular curve E sometimes such E' does not exist.

Empirical results To demonstrate the efficiency of this method, we present empirical results of our C++ implementation of FindCurve. We were looking for an elliptic curve over $\mathbb{F}_{2^{64}}$ with order divisible by 2^{34}. The only such number in the Hasse bound is 2^{64} and we found an elliptic curve of this order after checking 1803907700 curves in less then 8 hours on a single thread. Note that we came across significantly less than 2^{34} curves, that is because we checked only curves of the form E_B, which are known to have size divisible by 4. The curve that we found is $E_B : Y^2 + XY = X^3 + BX$ where B is a root of the following degree 64 irreducible polynomial over $\mathbb{F}_2$:

$$\begin{aligned}&X^{64} + X^{59} + X^{54} + X^{53} + X^{52} + X^{51} + X^{50} + X^{49} + X^{48} + X^{46} + X^{43} + X^{42}\\&+ X^{41} + X^{40} + X^{37} + X^{35} + X^{33} + X^{29} + X^{27} + X^{26} + X^{25} + X^{23} + X^{20}\\&+ X^{18} + X^{17} + X^{14} + X^{12} + X^9 + X^6 + X^4 + X^3 + X^2 + 1.\end{aligned}$$

5 Codes derived from the elliptic curves

In this section we construct two related families of linear codes, which are the main characters in our IOPP and STARK protocols. One family consists of AG codes, and the other of Reed–Solomon codes over specially chosen evaluation domains. AG codes are defined by three parameters: The curve that gives rise to the code; the divisor in the curve which defines a Riemann–Roch space of functions on the curve; and a set of points on the curve, disjoint from the divisor, on which the functions are evaluated.

We have already defined the relevant curves: these are the elliptic curves E_i discussed in the previous section. To describe the codes we still need to specify their divisors and evaluation domains. In the next subsections we will define these divisors and sets, building up to the codes themselves, and prove some useful properties which will be necessary for our protocols in the following sections.

5.1 Special sets in E_i and $\mathbb{F}_q$

In this subsection we define special sets $G_i, G_i' \subset E_i$ and $H_i \subset \mathbb{F}_q$, as well as polynomials $\Omega_i(X) \in \mathbb{F}_q[X]$, which will be necessary for defining the AG-code and relating it to the Reed–Solomon codes.

Let $E_0, \ldots, E_{\mathsf{k}-1}$ be a sequence of curves as in Section 4.2, with cyclic subgroups $\langle g_i \rangle \subset E_i$ of size $2^{\mathsf{k}-i}$ and good points $P_i = 2^{\mathsf{k}-i-2} g_i \in E_i$ for $i \leq \mathsf{k} - 2$. Denote $G_i = \langle g_i \rangle$ and define for each $i \leq \mathsf{k} - 1$

$$G_i' = G_i \setminus \{\mathbf{0}, \boldsymbol{\infty}\} = G_i \setminus \{2^{\mathsf{k}-i-1} g_i, 2^{\mathsf{k}-i} g_i\},$$

which is of size $2^{\mathsf{k}-i} - 2$. Denote $H_i = \pi_i(G_i')$, and note that $|H_i| = 2^{\mathsf{k}-i-1} - 1$, since $\pi_i : G_i' \to H_i$ is 2-to-1: indeed, the preimage of any $\pi_i(P) \in H_i$ is $\pi_i^{-1}(\pi_i(P)) = \{P, -P\}$, and $P \neq -P$ since the only 2-torsion points in G_i are $\mathbf{0}, \boldsymbol{\infty}$.

We can also characterize H_i by a reverse-recursive construction, starting with $H_{\mathsf{k}-1} = \emptyset$, and defining

$$H_i = \psi_i^{-1}(H_{i+1}) \cup \{b_i\}. \tag{10}$$

Indeed, every point $P = (x, y) \in G_i'$ either satisfies $\varphi_i(P) \in G_{i+1}'$, in which case $\psi_i(x) = \pi_{i+1}(\varphi_i(P)) \in H_{i+1}$, or else $\varphi(P) = \mathbf{0}$, i.e. $P = \pm P_i$ and $x = b_i$. This establishes

$$H_i \subseteq \psi_i^{-1}(H_{i+1}) \cup \{b_i\}.$$

In the other direction, note that ψ_i is at most 2-to-1, so

$$|\psi_i^{-1}(H_{i+1}) \cup \{b_i\}| \leq 2|H_{i+1}| + 1 = |H_i|.$$

This also implies that $\psi_i|_{H_i \setminus \{b_i\}} : H_i \setminus \{b_i\} \to H_{i+1}$ is exactly 2-to-1. It can also be directly checked that for every $x = \pi_i(\pm P) \in H_i \setminus \{b_i\}$, the point $x' = \frac{b_i^2}{x}$ with $\psi_i(x') = \psi_i(x)$ is also given by

$$x' = \pi_i(\pm P + \mathbf{0}) = \pi_i(\pm P + 2^{\mathsf{k}-i-1} g_i) \in H_i,$$

and indeed $\{\pm P + \mathbf{0}\}$ is disjoint from $\{\pm P\}$ whenever $P \neq \pm P_i$.

Define $\Omega_i(X) = \prod_{u \in H_i}(X - u)$, the vanishing polynomial of H_i, which is monic of degree $2^{\mathsf{k}-i-1} - 1$ (for $i = \mathsf{k}-1$, the empty product gives $\Omega_{\mathsf{k}-1} = 1$). Note that from (10) and $\psi_i(X)$ having denominator X we get for $i < \mathsf{k} - 1$ the recursive formula

$$\Omega_i(X) = (X - b_i) \cdot X^{2^{\mathsf{k}-i-2}-1} \cdot \Omega_{i+1}(\psi_i(X)). \tag{11}$$

5.2 Function and polynomial spaces

In this section we define three families of linear spaces, one consisting of polynomial over $\mathbb{F}_q$, another of rational functions over $\mathbb{F}_q$, and the third of functions on the elliptic curves E_i. The first and third will be the backbones of our Reed–Solomon and AG codes, respectively, whereas the second will act as a useful stepping stone between them. We will also define special bases for these spaces and prove several lemmas relating these spaces and bases to each other, which will be key to obtaining FFT and FRI equivalents for these functions and codes.

For any positive integer d, let

$$\mathbb{F}_q[X]^{<d} = \{p(X) \in \mathbb{F}_q[X] : \deg(p(X)) < d\}$$

denote the d-dimensional space of polynomials of degree strictly less than d. Denote $\mathcal{M}_i = \mathbb{F}_q[X]^{<2^{\mathsf{k}-i-1}}$ for every $i \leq \mathsf{k} - 1$. We define a related space of *rational* functions,

$$\mathcal{L}_i = \left\{ \frac{p(X)}{\Omega_i(X)} : p(X) \in \mathcal{M}_i \right\} \subset \mathbb{F}_q(X).$$

Clearly, $\mathcal{L}_i$ and $\mathcal{M}_i$ are isomorphic as linear spaces, with the isomorphism given by dividing or multiplying by $\Omega_i(X)$. In particular $\mathcal{L}_i$ is also of dimension $2^{\mathsf{k}-i-1}$. Another characterization of the elements of $\mathcal{L}_i$ is that they are the rational functions which have at most a simple pole at each $u \in H_i$, and no other poles (including at ∞). In other words, it is the Riemann–Roch space $\mathscr{L}([H_i])$.

Similarly to $\mathcal{L}_i$, let $\mathcal{K}_i = \mathscr{L}([G_i]) \subset k_{E_i}$ be the Riemann–Roch space with divisor $[G_i]$, i.e., the space of rational functions on E_i whose poles are all simple and lie in the subgroup G_i. By the Riemann–Roch theorem, $\dim \mathcal{K}_i = |G_i| = 2^{\mathsf{k}-i}$.

We now wish to connect between adjacent spaces $\mathcal{M}_i$ and $\mathcal{M}_{i+1}$; by the above-mentioned isomorphism, this will also be equivalent to a connection between adjacent $\mathcal{L}_i$ and $\mathcal{L}_{i+1}$. Additionally we would like to connect between $\mathcal{K}_i$ and $\mathcal{L}_i$, which is again equivalent to a connection between $\mathcal{K}_i$ and $\mathcal{M}_i$. In all of the above, we are connecting a certain space to a space of half its dimension; and indeed, these connections will all show that the larger space is the direct sum of two copies of the smaller space, suitably transformed, and these connections will give rise to recursively defined bases for the spaces.

5.2.1 Connections and bases for $\mathcal{M}_i$ and $\mathcal{L}_i$

To connect adjacent $\mathcal{M}_i$-s, we use the degree 2 rational functions ψ_i as well as two auxiliary linear functions:

Lemma 5.1. *Let $i \leq \mathsf{k} - 2$, and let $\chi_{i,0}, \chi_{i,1}$ be a basis of $\mathbb{F}_q[X]^{<2}$, i.e. any two independent linear polynomials. Denote $v_i(X) = X$, the denominator of ψ_i. Then*

$$\mathcal{M}_i = (\chi_{i,0} \cdot \psi_i^*(\mathcal{M}_{i+1}) \oplus \chi_{i,1} \cdot \psi_i^*(\mathcal{M}_{i+1})) \cdot v_i^{2^{\mathsf{k}-i-2}-1}. \tag{12}$$

In other words, any $f(X) \in \mathcal{M}_i$ can be uniquely decomposed as

$$f(X) = (\chi_{i,0}(X) f_0(\psi_i(X)) + \chi_{i,1}(X) f_1(\psi_i(X))) \cdot X^{2^{\mathsf{k}-i-2}-1}$$

with $f_0, f_1 \in \mathcal{M}_{i+1}$, and vice versa, every f of this form is in $\mathcal{M}_i$.

Proof. The lemma is an almost immediate application of [BCKL21, Lemma 3.1], with $d = 2^{\mathsf{k}-i-1}$ and $\delta = 2$. There is a slight difference in that the original statement was only for the standard basis $\chi_{i,0} = 1, \chi_{i,1} = X$ of $\mathbb{F}_q[X]^{<2}$. We leave it as an exercise for the reader to verify that the proof applies equally well over any basis; alternatively, that a decomposition in one basis can be transformed into a decomposition in any other basis by applying the corresponding base change transformation to the polynomials f_0, f_1. □

Next, we connect adjacent $\mathcal{L}_i$-s, again using ψ_i and an auxiliary function ξ_i, described below. This decomposition will be equivalent to the above decomposition of $\mathcal{M}_i$, under the identifications $\mathcal{L}_i = \Omega_i(X)^{-1}\mathcal{M}_i$, $\mathcal{L}_{i+1} = \Omega_{i+1}(X)^{-1}\mathcal{M}_{i+1}$; thus, the validity of the two decompositions will also be equivalent. For completeness, we will prove the validity of the $\mathcal{L}_i$ decomposition directly; thus the reader need not rely on [BCKL21, Lemma 3.1] for the proof of Lemma 5.1.

Lemma 5.2. *Let $i \leq \mathsf{k} - 2$, and let $\xi_i \in \mathbb{F}_q(X)$ be a rational function with a simple pole at b_i, and no other poles, including at ∞. Then*

$$\mathcal{L}_i = \psi_i^*(\mathcal{L}_{i+1}) \oplus \xi_i \cdot \psi_i^*(\mathcal{L}_{i+1}). \tag{13}$$

In other words, any $f \in \mathcal{L}_i$ can be uniquely decomposed as

$$f(X) = f_0(\psi_i(X)) + \xi_i(X) f_1(\psi_i(X))$$

with $f_0, f_1 \in \mathcal{L}_{i+1}$, and vice versa, every f of this form is in $\mathcal{L}_i$.

Proof. We first show that the sum on the right hand side of (13) is direct. Note that ψ_i is ramified at the points $\pm b_i$, and only there, and the ramification degree at those points is 2. It follows that every function in $\psi_i^*(\mathcal{L}_{i+1})$ has a zero or pole of even multiplicity at b_i. On the other hand, ξ_i has a simple pole at b_i, so any function in $\xi_i \cdot \psi_i^*(\mathcal{L}_{i+1})$ will have a zero or pole of odd multiplicity at b_i. In particular the two spaces are disjoint and the sum is direct.

It is now clear that both sides of (13) are of dimension $\dim \mathcal{L}_i = 2^{\mathsf{k}-i-1} = 2 \dim \mathcal{L}_{i+1}$, so it suffices to show that one contains the other in order to prove equality. Let us show that the right hand side is contained in the left hand side. As noted before, the only ramification points of ψ_i are at $\pm b_i$, but these points are not in $\psi_i^{-1}(H_{i+1})$, since they are x-coordinates of 4-torsion points, which φ_i maps either to $\mathbf{0}$ or outside G_{i+1}. It follows that for any functions $f_0, f_1 \in \mathcal{L}_{i+1}$, which have at most simple poles and only at the points of H_{i+1}, the functions $f_j \circ \psi_i$ will also have at most simple poles and only at the points of $\psi_i^{-1}(H_{i+1}) \subset H_i$, thus $f_0 \circ \psi_i \in \mathcal{L}_i$. Similarly, since ξ_i has only a simple pole at $b_i \notin \psi_i^{-1}(H_{i+1})$, $\xi_i \cdot (f_1 \circ \psi_i)$ will have at most simple poles at $\psi_i^{-1}(H_{i+1}) \cup \{b_i\} = H_i$ and again $\xi_i \cdot (f_1 \circ \psi_i) \in \mathcal{L}_i$. Thus $f_0 \circ \psi_i + \xi_i \cdot (f_1 \circ \psi_i) \in \mathcal{L}_i$ for any $f_0, f_1 \in \mathcal{L}_{i+1}$, as we wanted to show. □

Remark 5.3. To make the lemma more explicit, the condition on the poles of ξ_i is equivalent to it being equal to some linear function divided by $X - b_i$, such that the ratio is non-constant. We will see in Section 6.1 that it is also useful to have some nice connection between the values of $\xi_i(X)$ and $\xi_i\left(b_i^2/X\right)$, allowing for more efficient computations. For example, choosing $\xi_i(X) = \frac{X+b_i}{X-b_i}$ gives $\xi_i\left(b_i^2/X\right) = -\xi_i(X)$. This function only works for odd characteristic, since in characteristic 2 it is constant; there we may take instead $\xi_i(X) = \frac{b_i}{X-b_i}$, which satisfies $\xi_i\left(b_i^2/X\right) = \xi_i(X) + 1$ in characteristic 2.

Remark 5.4. As previously noted, Lemma 5.2 is equivalent to Lemma 5.1; more specifically, to the case where $\chi_{i,0} = X - b_i$ and $\chi_{i,1}$ is the numerator of ξ_i. The reader is invited to verify that the two decompositions can be deduced from each other by multiplying or dividing both sides by $\Omega_i(X)$, using the RHS of the recursion formula (11) for relating between $\mathcal{L}_{i+1}$ and $\mathcal{M}_{i+1}$.

Lemma 5.2 gives us constructions for natural bases $\{\lambda_{i,j}(X)\}_{j=0}^{2^{\mathsf{k}-i-1}-1}$ to each $\mathcal{L}_i$ and $\{\mu_{i,j}(X)\}_{j=0}^{2^{\mathsf{k}-i-1}-1}$ to $\mathcal{M}_i$:

Definition 5.5. *Let $\xi_0, \ldots, \xi_{\mathsf{k}-2}$ be as in Lemma 5.2. The base $\{\lambda_{\mathsf{k}-1,0}\}$ for $\mathcal{L}_{\mathsf{k}-1}$ is defined by $\lambda_{\mathsf{k}-1,0} = 1$, and for each $i = \mathsf{k}-2, \ldots, 0$, the basis $\{\lambda_{i,j}(X)\}_{j=0}^{2^{\mathsf{k}-i-1}-1}$ is defined recursively by*

$$\lambda_{i,2j} = \lambda_{i+1,j} \circ \psi_i, \quad \lambda_{i,2j+1} = \xi_i \cdot \lambda_{i+1,j} \circ \psi_i.$$

The bases $\{\mu_{i,j}(X)\}_{j=0}^{2^{\mathsf{k}-i-1}-1}$ to $\mathcal{M}_i$ are then defined by $\mu_{i,j} = \Omega_i \cdot \lambda_{i,j}$.

Note that the $\mu_{i,j}$ defined above also satisfy recursion relations similar to Lemma 5.1, specifically,

$$\mu_{i,2j} = X^{2^{\mathsf{k}-i-2}-1} \cdot \chi_{i,0} \cdot \mu_{i+1,j} \circ \psi_i, \quad \mu_{i,2j+1} = X^{2^{\mathsf{k}-i-2}-1} \cdot \chi_{i,1} \cdot \mu_{i+1,j} \circ \psi_i$$

where $\chi_{i,0} = X - b_i$, $\chi_{i,1} = (X - b_i) \cdot \xi_i$,

5.2.2 Connections and bases for $\mathcal{K}_i$

Next, we decompose $\mathcal{K}_i$ as the sum of two copies of $\mathcal{L}_i$:

Lemma 5.6. *Let $i \leq \mathsf{k}-1$, and let $\zeta_i(X,Y) = \frac{Y}{X}$. Then*

$$\mathcal{K}_i = \pi_i^*(\mathcal{L}_i) \oplus \zeta_i \cdot \pi_i^*(\mathcal{L}_i).$$

In other words, any $f(X,Y) \in \mathcal{K}_i$ can be uniquely decomposed as

$$f(X,Y) = f_0(\pi_i(X,Y)) + \zeta_i(X,Y) f_1(\pi_i(X,Y)) = f_0(X) + \frac{Y}{X} f_1(X) \tag{14}$$

with $f_0, f_1 \in \mathcal{L}_i$, and vice versa, every f of this form is in $\mathcal{K}_i$.

Proof. Since Y^2 can be written as a function of X on the curve, it is immediate and well known that any rational function $f(X,Y)$ can be decomposed uniquely as in (14), with f_0, f_1 rational functions. Thus it only remains to be seen that given such a decomposition, f is in $\mathcal{K}_i$ if and only if f_0, f_1 are both in $\mathcal{L}_i$.

In one direction, suppose $f_0, f_1 \in \mathcal{L}_i$. Since $\mathcal{K}_i$ is a linear space, it is enough to check that $f_0(X)$ and $\frac{Y}{X} f_1(X)$ are both in $\mathcal{K}_i$. By definition of $\mathcal{L}_i$ we know that as rational functions in one variable, f_0, f_1 can have at most simple poles and only at points of H_i. It follows that $f_0 \circ \pi_i$, $f_1 \circ \pi_i$ can have poles only at the points of $\pi_i^{-1}(H_i) = G_i' \subset G_i$, and indeed these poles are all simple since $\pi_i|_{G_i'} : G_i' \to H_i$ is exactly 2-to-1 and has no ramified points. In particular $f_0 \circ \pi_i \in \mathcal{K}_i$. Furthermore, the function $\zeta_i = \frac{Y}{X}$ has exactly two simple poles, one at each point of $G_i \setminus G_i' = \{\mathbf{0}, \boldsymbol{\infty}\}$. Since $f_1 \circ \pi_i$ does not have poles at these points, it follows that $\zeta_i \cdot f_1 \circ \pi_i$ also has at most simple poles at the points of G_i, i.e. belongs to $\mathcal{K}_i$, as claimed.

From the above analysis it follows that the map

$$(f_0, f_1) \mapsto f_0 \circ \pi_i + \zeta_i \cdot f_1 \circ \pi_i$$

is a linear map from $\mathcal{L}_i \oplus \mathcal{L}_i$ to $\mathcal{K}_i$. The uniqueness of the (general) decomposition implies this map is injective. Since we also have $\dim \mathcal{K}_i = 2^{\mathsf{k}-i} = \dim(\mathcal{L}_i \oplus \mathcal{L}_i)$, the map must be an isomorphism, and in particular surjective, i.e. the required decomposition indeed exists. □

Remark 5.7. As in Lemma 5.2 and Remark 5.3, one may replace $\frac{Y}{X}$ by a different function ζ_i, as long as it has the same poles, which is equivalent to it being a linear combination of $\frac{Y}{X}$ and a constant function. The particular choice of $\zeta_i = \frac{Y}{X}$ will provide elegant formulas in both odd and even characteristic.

Using the basis we have for $\mathcal{L}_i$, Lemma 5.6 also yields a canonical basis $\{\kappa_{i,j}\}_{j=0}^{2^{\mathsf{k}-i}-1}$ to $\mathcal{K}_i$:

Definition 5.8. *For each $i = 0, \ldots, \mathsf{k}-1$, the basis $\{\kappa_{i,j}\}_{j=0}^{2^{\mathsf{k}-i}-1}$ is defined by*

$$\kappa_{i,2j} = \lambda_{i,j} \circ \pi_i,\ \kappa_{i,2j+1} = \zeta_i \cdot \lambda_{i,j} \circ \pi_i.$$

As a corollary of Lemma 5.6, we can also decompose $\mathcal{K}_i$ as the sum of two copies of $\mathcal{M}_i$, simply by including an additional factor of Ω_i:

Corollary 5.9. *Let $i \leq \mathsf{k}-1$, and let $\zeta_i(X,Y) = \frac{Y}{X}$. Then*

$$\mathcal{K}_i = (\pi_i^*(\mathcal{M}_i) \oplus \zeta_i \cdot \pi_i^*(\mathcal{M}_i)) \cdot (\Omega_i \circ \pi_i)^{-1}.$$

In other words, any $f(X,Y) \in \mathcal{K}_i$ can be uniquely decomposed as

$$f(X,Y) = \frac{f_0(\pi_i(X,Y)) + \zeta_i(X,Y) f_1(\pi_i(X,Y))}{\Omega_i(X)} = \frac{f_0(X)}{\Omega_i(X)} + \frac{Y}{X}\frac{f_1(X)}{\Omega_i(X)} \tag{15}$$

with $f_0, f_1 \in \mathcal{M}_i$, and vice versa, every f of this form is in $\mathcal{K}_i$. □

Remark 5.10. The curious reader might ask: If $\mathcal{M}_i$ and $\mathcal{L}_i$ are direct sums of copies of $\psi_i^*(\mathcal{M}_i)$ and $\psi_i^*(\mathcal{L}_i)$, correspondingly, shouldn't $\mathcal{K}_i$ also be a direct sum of copies of $\varphi_i^*(\mathcal{K}_{i+1})$? Shouldn't there be such a decomposition which commutes with the decompositions of Lemmas 5.2 and 5.6? Perhaps surprisingly, the answer is no. The reader is invited to check that the only functions η_i such that $\eta_i \cdot \varphi_i^*(\mathcal{K}_{i+1}) \subset \mathcal{K}_i$ are the constant functions: η_i may not have any poles, because for every point in E_i there exists functions in $\varphi_i^*(\mathcal{K}_{i+1})$ with the maximal allowed order of pole at the point (i.e. 1 if the point is in G_i or 0 otherwise). Thus, any sum $\eta_{i,0} \cdot \varphi_i^*(\mathcal{K}_{i+1}) + \eta_{i,1} \cdot \varphi_i^*(\mathcal{K}_{i+1})$ is either not direct or not wholly contained in $\mathcal{K}_i$, and in any case is not equal to it. Attempting to construct such a decomposition so as to commute with those of Lemmas 5.2 and 5.6 fails because ζ_i does not equal $\zeta_{i+1} \circ \varphi_i$, and in fact is not in $\varphi_i^*(k_{E_{i+1}})$ at all.

5.3 Evaluation domains

In this subsection we construct the evaluation domains used for our Reed–Solomon and AG codes.

Let Q_0 be a point on E_0 such that $2Q_0$ is not in $2G_0 = \langle 2g_0 \rangle$. Define the *basic set for G_0 corresponding to Q_0* as

$$S_0 = S_0(Q_0) = (Q_0 + \langle 2g_0 \rangle) \cup (-Q_0 + \langle 2g_0 \rangle).$$

Because $2Q_0 \notin 2G_0$, this is a union of two distinct, complementary cosets of $2G_0$, and is of size 2^{k}. Define inductively for $0 \leq i \leq \mathsf{k}-2$, $Q_{i+1} = \varphi_i(Q_i)$. Note that $2Q_{i+1} = \varphi_i(2Q_i) \notin 2G_{i+1}$, since $2Q_i \notin 2G_i = \varphi_i^{-1}(2G_{i+1})$, and therefore

$$S_{i+1} = \varphi_{i+1}(S_i) = (Q_{i+1} + \langle 2g_{i+1} \rangle) \cup (-Q_{i+1} + \langle 2g_{i+1} \rangle)$$

is again a union of two complementary and distinct cosets of $2G_{i+1}$, and of size $2^{\mathsf{k}-i-1}$.

Define $T_i = \pi_i(S_i)$ for $0 \leq i \leq \mathsf{k}-1$, and note that $|T_i| = 2^{\mathsf{k}-i-1}$, since $\pi_i|_{S_i}$ is exactly 2-to-1, with all fibers being of the form $\{Q_i + 2kg_i, -Q_i - 2kg_i\}$. By the commutativity of Eq. (3) we also get that $T_{i+1} = \psi_i(T_i)$, with $\psi_i|_{T_i}$ being 2-to-1. We note that it is easy to to split T_i to the pairs corresponding to fibers of ψ_i, i.e. pairs $\{x, x'\} \subset T_i$, such that $\psi_i(x) = \psi_i(x')$ (or equivalently, $x' = \frac{b_i^2}{x}$): indeed, if $x = x_j = \pi_i(Q_i + 2jg_i)$, then it will be paired with $x' = x_{j+2^{\mathsf{k}-i-2}} = \pi_i(Q_i + 2(j + 2^{\mathsf{k}-i-2})g_i)$, since

$$\begin{aligned}
&\varphi_i(Q_i + 2(j + 2^{\mathsf{k}-i-2})g_i) = \varphi_i(Q_i + 2jg_i) + \varphi_i(2^{\mathsf{k}-i-1}g_i) \\
&= \varphi_i(Q_i + 2jg_i) + \infty = \varphi_i(Q_i + 2jg_i) \\
\Rightarrow\ &\psi_i(\pi_i(Q_i + 2(j + 2^{\mathsf{k}-i-2})g_i)) = \pi_{i+1}(\varphi_i(Q_i + 2(j + 2^{\mathsf{k}-i-2})g_i))
\end{aligned}$$

$$= \pi_{i+1}(\varphi_i(Q_i + 2jg_i)) = \psi_i(\pi_i(Q_i + 2jg_i)).$$

Note that the S_i are disjoint from G_i, and thus similarly T_i are disjoint from H_i. It follows that for any function $f \in \mathcal{K}_i$ we can evaluate $f(s) \in \mathbb{F}_q$ for all $s \in S_i$, and likewise for $f \in \mathcal{L}_i$ we can evaluate $f(t) \in \mathbb{F}_q$ for all $t \in T_i$, since such functions are guaranteed to not have poles in S_i or T_i, respectively. Similarly, the polynomial $\Omega_i(t)$ is non-zero for any $t \in T_i$.

Our evaluation domains will be made from unions of disjoint basic sets. For a set of points $\mathbf{Q}_0 = \{Q_{0,1}, \ldots, Q_{0,m}\} \subset E_0$ such that the corresponding basic sets are all pairwise disjoint, let

$$\mathbf{S}_0 = \bigcup_{k=1}^{m} S_{0,k} = \bigcup_{k=1}^{m} S_0(Q_{0,k}), \quad \mathbf{T}_0 = \bigcup_{k=1}^{m} T_{0,k} = \bigcup_{k=1}^{m} \pi_0(S_{0,k})$$

and again define $\mathbf{Q}_i, \mathbf{S}_i, \mathbf{T}_i$ using φ_i, π_i, ψ_i. We will have $|\mathbf{S}_i| = 2^{\mathsf{k}-i} \cdot m$, $|\mathbf{T}_i| = 2^{\mathsf{k}-i-1} \cdot m$.

Following the notation in [BCKL21], for a function f defined on an evaluation domain S, we denote by $\langle f \wr S \rangle$ the *evaluation table* of f on S.

Remark 5.11. Readers might recognize the construction of the sets T_i from [BCKL21]: these correspond exactly to the construction of an FFTree, with $T_i(Q_i)$ being its i-th layer. The new notions here are the construction of the sets S_i, which can be considered as an extension of the FFTree to the curves rather than just $\mathbb{F}_q$; and considering multiple such trees side-by-side in $\mathbf{T}_i$ and $\mathbf{S}_i$, useful for error-correcting codes.

5.4 The codes

We are finally ready to introduce our families of codes, mentioned earlier:

Definition 5.12 (Elliptic Curve Codes)**.** *Given a curve E_i and an evaluation domain $\mathbf{S}_i$ as above, denote*

$$U_i = U_i(\mathbf{S}_i) = C(\mathbf{S}_i, [G_i]) = \{\langle f \wr \mathbf{S}_i \rangle : f \in \mathcal{K}_i\}.$$

This is the AG code over E_i with divisor $[G_i]$ and evaluation domain $\mathbf{S}_i$, which has blocksize $|\mathbf{S}_i| = 2^{\mathsf{k}-i} \cdot m$, dimension $\dim \mathcal{K}_i = 2^{\mathsf{k}-i}$ and rate $\frac{1}{m}$.

Similarly, denote

$$W_i = \mathsf{RS}[\mathbb{F}_q, \mathbf{T}_i, \tfrac{1}{m}] = \{\langle f \wr \mathbf{T}_i \rangle : f \in \mathcal{M}_i\},$$

which is a Reed–Solomon code of blocksize $|\mathbf{T}_i| = 2^{\mathsf{k}-i-1} \cdot m$, dimension $\dim \mathcal{M}_i = 2^{\mathsf{k}-i-1}$ and rate $\frac{1}{m}$.

The two codes are deeply related: Corollary 5.9 gives us a natural isomorphism between U_i and $W_i \oplus W_i$, which will be discussed further in Section 6.4. It will also be useful to keep in mind the AG code

$$V_i = C(\mathbf{T}_i, [H_i]) = \{\langle f \wr \mathbf{T}_i \rangle : f \in \mathcal{L}_i\},$$

defined over the projective line $\mathbb{P}^1$, as an intermediary between U_i and W_i. The isomorphism between $\mathcal{L}_i$ and $\mathcal{M}_i$ yields an isomorphism between V_i and W_i, via pointwise multiplication or division by $\langle \Omega_i \wr \mathbf{T}_i \rangle$ (which is non-zero valued).

An important property of the code U_i is that it has a large *automorphism group*, and more specifically that there is a large *cyclic* group of code automorphisms. An automorphism of a code is a permutation of the words' entries which preserves the property of being a codeword[12]. These automorphisms can be described explicitly: For any point $P \in \mathbf{E}_i$, let $\tau_P : E_i \to E_i$ denote the translation by P map, defined by $\tau_P(Q) = Q + P$, $\forall Q \in E_i$. Note that for any $P \in G_i$, $\tau_P(G_i) = G_i$, which implies that $\tau_P^* : f \mapsto f \circ \tau_P$ is an automorphism of $\mathcal{K}_i$, as simple poles in G_i are shifted

[12]The definition of an automorphism sometimes allows also an additional coordinate-wise scaling before or after the permutation; this will not be relevant for our purposes.

to simple poles in G_i. If we further restrict to $P \in 2G_i$, we find that the map $\tau_P^* : \langle f \wr \mathbf{S}_i \rangle \mapsto \langle f \circ \tau_P \wr \mathbf{S}_i \rangle$ is indeed an automorphism of U_i, where the entries are permuted inside the $2m$ different cosets of $2G_i$ which comprise $\mathbf{S}_i$; the different cosets are never intermixed by these automorphisms. Finally, note that $\tau : P \mapsto \tau_P$ is itself a homomorphism, i.e. $\tau_{P+Q} = \tau_P \circ \tau_Q$, and thus so is $\tau^* : P \mapsto \tau_P^*$, It follows that $\tau^*_{2G_i} = \langle \tau^*_{2g_i} \rangle$ is a cyclic group of automorphisms of U_i of size $2^{\mathsf{k}-i-1}$. Equivalently, this can be viewed as a faithful action of $2G_i$ on U_i by code automorphisms.

Remark 5.13. There are additional natural automorphisms of the code U_i, given by composition with reflection maps $Q \leftrightarrow P - Q$, for $P \in 2G_i$. These automorphisms no longer preserve the cosets of $2G_i$ comprising $\mathbf{S}_i$, but rather switch between the two pairs of cosets in each basic sets $S_{i,k}$. These reflection maps, together with the translations τ_{2G_i}, form a *dihedral* group of size $2^{\mathsf{k}-i}$; equivalently we can say that this dihedral group acts on E_i and on U_i. In fact, it is more natural to define the basic sets $S_{i,k}$ as *orbits* of this dihedral action. For our purposes, the cyclic part of the action is more useful than the full dihedral action, and the reader can safely ignore these extra automorphisms.

6 ECFFT and ECFRI

In this section we examine FFT-like algorithms for the spaces $\mathcal{K}_i, \mathcal{L}_i, \mathcal{M}_i$, which transform between their representations in the bases defined in Section 5.2 and their evaluations on the basic sets defined in Section 5.3 in quasilinear time, as described in Theorem 6.1. We then present several applications of these algorithms: In Section 6.2 we describe low degree extensions, which allows for fast generation of the codewords from codes defined in Section 5.4. In Section 6.3 we give a high level description of fast IOPP for these same curves. Finally, in Section 6.4 we describe in detail a certain relationship between the codes which arises from the FFT structure.

Some of the results in this section, particularly in Section 6.2, are retreading of ideas appearing in [BCKL21]. This section generalizes and expands on these ideas, to more general function spaces and further applications.

6.1 ECFFT

Recall that the function spaces $\mathcal{K}_i, \mathcal{L}_i, \mathcal{M}_i$ have special bases $\kappa_{i,j}$, $\lambda_{i,j}$, $\mu_{i,j}$ (Definitions 5.5 and 5.8). For a function $f \in \mathcal{K}_i$, let $[f]_{\kappa_i}$ represent its representation in the κ_i basis, i.e. the vector $(c_j)_{j=0,\ldots,2^{\mathsf{k}-i}-1}$ such that $f = \sum c_j \kappa_{i,j}$. Similarly, we will denote by $[f]_{\lambda_i}$ the representation of a function $f \in \mathcal{L}_i$ in the λ_i basis, and by $[f]_{\mu_i}$ the representation of a function $f \in \mathcal{M}_i$ in the μ_i basis.

Let $Q_i \in E_i$ generate an orbit S_i of size $2^{\mathsf{k}-i}$ and let $T_i = \pi_i(S_i)$.

Theorem 6.1 (ECFFT)**.** *There exist invertible linear transformations* $\mathsf{FFT}_{\mathcal{K}_i,S_i}$, $\mathsf{IFFT}_{\mathcal{K}_i,S_i}$ *such that* $\forall f \in \mathcal{K}_i$,

$$\mathsf{FFT}_{\mathcal{K}_i,S_i}([f]_{\kappa_i}) = \langle f \wr S_i \rangle\,,\ \mathsf{IFFT}_{\mathcal{K}_i,S_i}(\langle f \wr S_i \rangle) = [f]_{\kappa_i}.$$

Similarly, there exist invertible linear transformations $\mathsf{FFT}_{\mathcal{L}_i,T_i}$, $\mathsf{IFFT}_{\mathcal{L}_i,T_i}$ *such that* $\forall f \in \mathcal{L}_i$,

$$\mathsf{FFT}_{\mathcal{L}_i,T_i}([f]_{\lambda_i}) = \langle f \wr T_i \rangle\,,\ \mathsf{IFFT}_{\mathcal{L}_i,T_i}(\langle f \wr T_i \rangle) = [f]_{\lambda_i},$$

and invertible linear transformations $\mathsf{FFT}_{\mathcal{M}_i,T_i}$, $\mathsf{IFFT}_{\mathcal{M}_i,T_i}$ *such that* $\forall f \in \mathcal{M}_i$,

$$\mathsf{FFT}_{\mathcal{M}_i,T_i}([f]_{\mu_i}) = \langle f \wr T_i \rangle\,,\ \mathsf{IFFT}_{\mathcal{M}_i,T_i}(\langle f \wr T_i \rangle) = [f]_{\mu_i}.$$

Moreover, all $\mathsf{FFT}, \mathsf{IFFT}$ *above are computable in* $O(N \log_2 N)$ *arithmetic operations over the ambient field, where* N *is the dimension of the space (which equals the size of the evaluation domain).*

Proof. We first prove for $\mathcal{L}_i, T_i$. For $i = \mathsf{k}-1$, $\mathcal{L}_i$ consists only of constant functions, with $[c] = (c)$ and $\langle c \wr T_i \rangle = (c)$, so the transforms are simply the identity. We continue by reverse induction. For any $f \in \mathcal{L}_i$, $i \leq \mathsf{k} - 2$, write

$$f(x) = f_0(\psi_i(x)) + \xi_i(x) f_1(\psi_i(x)) \tag{16}$$

as in Lemma 5.2, with $f_0, f_1 \in \mathcal{L}_{i+1}$. Note that by the definition of the base λ_i from λ_{i+1}, the translation between $[f]_{\lambda_i}$ and $[f_0]_{\lambda_{i+1}}, [f_1]_{\lambda_{i+1}}$ is trivial — the coefficient of $\lambda_{i+1,j}$ in $[f_k]_{\lambda_{i+1}}$ is exactly the coefficient of $\lambda_{i,2j+k}$ in $[f]_{\lambda_i}$. Using $\mathsf{FFT}_{T_{i+1}}$ or $\mathsf{IFFT}_{T_{i+1}}$, we can replace $[f_0]_{\lambda_{i+1}}, [f_1]_{\lambda_{i+1}}$ by $\langle f_0 \wr T_{i+1} \rangle, \langle f_1 \wr T_{i+1} \rangle$ (or vice versa) in $O(2|T_{i+1}| \log_2 |T_{i+1}|)$ operations. To conclude, we need to show that $\langle f_0 \wr T_{i+1} \rangle, \langle f_1 \wr T_{i+1} \rangle$ and $\langle f \wr T_i \rangle$ can be interchanged in $O(|T_i|)$ operations. And this is indeed doable: let $x_0, x_1 \in T_i$ be any pair with $\psi_i(x_0) = \psi_i(x_1) = x'$. As noted in Section 5.3, such pairs satisfy $x_0 x_1 = b_i^2$, so $x' = x_0 + x_1 - 2b_i$, and they can be easily located as they are always at distance $\frac{1}{2}|T_i|$ from each other when ordered according to the coset. Substituting $x = x_j$ in (16), we find

$$f(x_0) = f_0(x') + \xi_i(x_0) f_1(x'), \quad f(x_1) = f_0(x') + \xi_i(x_1) f_1(x'). \tag{17}$$

Since ξ_i is a degree-1 rational function, it is one-to-one, thus $\xi_i(x_0) \neq \xi_i(x_1)$ and the equation system (17) is invertible, allowing to solve for $f_0(x'), f_1(x')$ from $f(x_0), f(x_1)$. When $\xi_i(x_0), \xi_i(x_1)$ are nicely related, the inversion formula can also be simplified: If $\xi_i(x_1) = -\xi_i(x_0)$ (in the odd characteristic case), then

$$f_0(x') = \frac{f(x_0) + f(x_1)}{2}, \quad f_1(x') = \frac{f(x_0) - f(x_1)}{2\xi_i(x_0)}; \tag{18}$$

and in characteristic 2, if $\xi_i(x_1) = \xi_i(x_0) + 1$, then

$$f_1(x') = f(x_0) + f(x_1), \quad f_0(x') = f(x_0) + \xi_i(x_0) f_1(x'). \tag{19}$$

We can thus exchange between $f(x_0), f(x_1)$ and $f_0(x'), f_1(x')$ in $O(1)$ operations, using one of (17), (18) or (19) depending on the direction and characteristic. More specifically, assuming precomputation of values of the $\xi_i(x_0)$ and their inverses, we use only 1 multiplication and 2 additions/subtractions for each pair, in either direction (for IFFT in odd characteristic, we can postpone all divisions by 2 to the end). We repeat this $\frac{1}{2}|T_i|$ times to cover all values, to obtain the transition in $O(|T_i|)$ operations, as claimed.

For functions in $\mathcal{K}_i$, the argument is similar to the argument for $\mathcal{L}_i$, using the decomposition

$$f(x, y) = f_0(x) + \zeta_i(x, y) f_1(x) \tag{20}$$

from Lemma 5.6, with $f_0, f_1 \in \mathcal{L}_i$. Again the translation between $[f]_{\kappa_i}$ and $[f_0]_{\lambda_i}, [f_1]_{\lambda_i}$ is trivial by the choice of the base, the coefficients $[f_j]_{\lambda_i}$ can be translated to and from values on T_i using $\mathsf{FFT}_{\mathcal{L}_i, T_i}$ and $\mathsf{IFFT}_{\mathcal{L}_i, T_i}$, and for any pair $(x, y), (x, y') \in S_i$ (where $y' = -y$ in odd characteristic, or $y' = y + x$ in characteristic 2) we use the relations

$$f(x, y) = f_0(x) + \zeta_i(x, y) f_1(x), \quad f(x, y') = f_0(x) + \zeta_i(x, y') f_1(x), \tag{21}$$

for the remaining transitions in FFT. Similarly to the above for ξ_i, for $\zeta_i(X, Y) = \frac{Y}{X}$ we again have $\zeta_i(x, y') = -\zeta_i(x, y)$ in odd characteristic and $\zeta_i(x, y') = \zeta_i(x, y) + 1$ in even characteristic, so to complete the IFFT we may use either of

$$f_0(x) = \frac{f(x, y) + f(x, -y)}{2}, \quad f_1(x) = \frac{f(x, y) - f(x, -y)}{2\zeta_i(x, y)}, \text{ or} \tag{22}$$

$$f_1(x) = f(x, y) + f(x, y + x), \quad f_0(x) = f(x, y) + \zeta_i(x, y) f_1(x) \tag{23}$$

depending on the parity of the characteristic.

For $\mathcal{M}_i, T_i$, we may argue similarly to the $\mathcal{L}_i$ case using Lemma 5.1, with the forward formulas

$$f(x_0) = x_0^{2^{k-i-1}-1} \cdot (\chi_{i,0}(x_0) f_0(x') + \chi_{i,1}(x_0) f_1(x')), \tag{24}$$

$$f(x_1) = x_1^{2^{k-i-1}-1} \cdot (\chi_{i,0}(x_1) f_0(x') + \chi_{i,1}(x_1) f_1(x')), \tag{25}$$

and inverse formulas similarly involving more coefficients; thus we may need to perform up to 4 multiplications instead of only one at every level of the transform; the coefficients themselves are assumed to be precomputed and given as advice.

A more efficient way to compute $\mathsf{FFT}_{\mathcal{M}_i,T_i}$ is by instead running $\mathsf{FFT}_{\mathcal{L}_i,T_i}$ on the same input (interpreted as coefficients in λ_i), and multiplying the result pointwise by precomputed values of $\langle \Omega_i \wr T_i \rangle$, in $|T_i|$ operations. This yields the correct result, since $\mu_{i,j} = \Omega_i \cdot \lambda_{i,j}$. Similarly, $\mathsf{IFFT}_{\mathcal{M}_i,T_i}$ can be computed by first dividing by $\langle \Omega_i \wr T_i \rangle$ and then running $\mathsf{IFFT}_{\mathcal{L}_i,T_i}$. □

These FFTs and their inverses are analogous to the usual FFT. The main difference is that for standard FFT, the basis for the space of polynomials is the standard basis, so $[f]$ is simply the vector of coefficients of monomials, whereas in the EC case the "natural" basis is more complicated. The circuit itself is very similar, consisting of $\frac{N}{2}$ butterflies in each of the $\log_2 N$ layers, each layer using a different stride size. For the transforms of $\mathcal{K}_i$ and $\mathcal{L}_i$ in the odd characteristic case, the only difference between these butterflies and those of the usual FFT is in the twiddle factors used: the values of ζ_i or ξ_i at the points of S_i or T_i, instead of roots of unity (or a coset). These values are determined by a precomputation, which the FFT/IFFT circuit need not be aware of.

6.2 Low Degree Extensions and the codes

Let $\mathbf{S}_i = \bigcup_{k=1}^{m} S_{i,k}$ be an evaluation domain as in Section 5.3. Given an array of elements of $\mathbb{F}_q$ of size $2^{\mathsf{k}-i}$, we can interpret it as an evaluation table $\langle f \wr S_{i,1} \rangle$ of a unique function $f \in \mathcal{K}_i$, and compute $[f]_{\kappa_i}$ by $\mathsf{IFFT}_{\mathcal{K}_i,S_{i,1}}$. Then, by applying $\mathsf{FFT}_{\mathcal{K}_i,S_{i,2}}, \ldots, \mathsf{FFT}_{\mathcal{K}_i,S_{i,m}}$ to $[f]_{\kappa_i}$, we can evaluate $\langle f \wr \mathbf{S}_i \rangle \in U_i$. This process is the *low degree extension* (or LDE) from $\langle f \wr S_{i,1} \rangle$ to $\langle f \wr \mathbf{S}_i \rangle$, which extends a given array into the unique U_i codeword that matches it.

Similarly, let $\mathbf{T}_i = \bigcup_{k=1}^{m} T_{i,k}$ be an evaluation domain for $\mathcal{L}_i$ or $\mathcal{M}_i$. We can uniquely extend any function $\langle f \wr T_{i,1} \rangle$ to $\langle f \wr \mathbf{T}_i \rangle$ with $f \in \mathcal{L}_i$ or $f \in \mathcal{M}_i$, yielding codewords in V_i or W_i respectively, and again the extensions can be computed efficiently with inverse and forward FFTs.

In [BCKL21], the main building block of the algorithms was the EXTEND algorithm, transforming an evaluation table of a low degree polynomial $\langle P \wr T_{i,0} \rangle$ to an evaluation of it on a different set $\langle P \wr T_{i,1} \rangle$. It can be seen that this algorithm indeed simply the composition of $\mathsf{FFT}_{\mathcal{M}_i,T_{i,1}}$ on $\mathsf{IFFT}_{\mathcal{M}_i,T_{i,0}}$, as was briefly mentioned in the paper. The reader is encouraged to compare the EXTEND algorithm as described in [BCKL21] to this composition of FFTs and observe they are equivalent.

6.3 EC-FRI

In this section we present a high-level description of how the FFT structure enables the FRI protocol for the Reed–Solomon codes W_i, and similar results for the AG-codes U_i and V_i. More details about this protocol, including the choice of the queries and the soundness analysis, appear in Appendix B.

Let $\mathbf{T}_0 = \bigcup_{k=1}^{m} T_{0,k}$ be an evaluation domain for $\mathcal{M}_0$ as in Section 5.3, and let $T_{i,k}, \mathbf{T}_i$ be the corresponding domains for $\mathcal{M}_i$. A prover P has access to a function $f_0 : \mathbf{T}_0 \to \mathbb{F}_q$, and wishes to prove to verifier V that it is an evaluation table $\langle f \wr \mathbf{T}_0 \rangle$ of a function $f \in \mathcal{M}_0$, i.e. a codeword from U_0, or at least close to one in Hamming distance.

Over $\mathsf{k}-1$ (or fewer) rounds of interaction, the verifier will provide randomness z_i, and the prover will commit (and provide oracle access to) a table $\langle f_{i+1} \wr \mathbf{T}_{i+1} \rangle$, where f_{i+1} is supposedly defined by

$$f_{i+1}(X) = f_{i,0}(X) + z_i f_{i,1}(X) \tag{26}$$

where $f_{i,0}, f_{i,1} \in M_{i+1}$ are the functions from the decomposition

$$f_i(X) = (\chi_{i,0}(X) f_{i,0}(\psi_i(X)) + \chi_{i,1}(X) f_{i,1}(\psi_i(X))) \cdot X^{2^{\mathsf{k}-i-2}-1} \tag{27}$$

given by Lemma 5.1. The verifier can check that f_{i+1} was appropriately computed by querying f_{i+1} at points $x' \in \mathbf{T}_{i+1}$ and f_i at $\{x_0, x_1\} = \psi_i^{-1}(x') \subset \mathbf{T}_i$, computing $f_{i,0}(x'), f_{i,1}(x')$ from $f_i(x_0), f_i(x_1)$ by (27) (as in the matching $\mathsf{IFFT}_{\mathcal{M}_i}$), and verifying (26) holds at x'. In the final round the prover will also commit to $f_{\mathsf{k}-1}$ being a constant

function; or, if performing only r rounds, will give f_r explicitly, for example by the coefficient list $[f_r]_{\mu_r}$, allowing for quick evaluation to each $T_{r,k}$.

A similar method can be used for proving proximity to the V_0 code, i.e. to an evaluation table $\langle f \wr \mathbf{T}_0 \rangle$ with $f \in \mathcal{L}_0$. The process is equivalent to that for $\mathcal{M}_i$ and the U_i codes above, and in each step both the words and the codes differ only by fixed pointwise multiplications, which does not affect the Hamming distance. Note that verifying the relationship between f_i and f_{i+1} in this setting will require the prover and the verifier to use the $\mathsf{IFFT}_{\mathcal{L}_i}$ formulas (either (18) or (19) depending on the characteristic) instead, which are computationally cheaper than inverting (26).

We can extend this method to an IOPP for the AG code U_0: Given an evaluation domain $\mathbf{S}_0 = \bigcup_{k=1}^{m} S_{0,k}$ for $\mathcal{K}_0$ and an evaluation table $\langle h \wr \mathbf{S}_0 \rangle$ supposedly of a function $h \in \mathcal{K}_0$, the prover can either compute the decomposition

$$h(X,Y) = \widetilde{h}_0(X) + \zeta_0(X,Y)\widetilde{h}_1(X) \tag{28}$$

from Lemma 5.6 or

$$h(X,Y) = \frac{h_0(X)}{\Omega_0(X)} + \zeta_0(X,Y)\frac{h_1(X)}{\Omega_0(X)} \tag{29}$$

from Corollary 5.9. The prover then computes $f_0 = \widetilde{h}_0 + z \cdot \widetilde{h}_1$ (or $h_0 + z \cdot h_1$) and commits to $\langle f_0 \wr \mathbf{T}_0 \rangle$, where z is randomness from the verifier. After this first step the protocol continues as above to show f_0 is close to a V_0/W_0 codeword. As before, the methods are equivalent whether we work with V_i or W_i, but the V_i method is more efficient. However, we will focus on working over W_i in the next section, to more immediately apply existing soundness results.

6.4 The isomorphism between U_0 and $W_0 \oplus W_0$

We note that equation (29), evaluated at all points $(x,y) \in \mathbf{S}_0$, gives rise to an isomorphism between U_0 and $W_0 \oplus W_0$, which sends $\langle h \wr \mathbf{S}_0 \rangle$ to the table of pairs $\langle (h_0, h_1) \wr \mathbf{T}_0 \rangle$ and vice versa; the same equation and map can also be applied to evaluation tables or pairs which are not necessarily codewords. Importantly, this map is local: to access $h_0(x)$ and/or $h_1(x)$, h need only be accessed at the two points in $\{(x,y),(x,y')\} = \pi_0^{-1}(x)$. This also implies that this isomorphism roughly preserves the Hamming distance; more specifically, if we consider U_0 and $W_0 \oplus W_0$ as codes over the alphabet $\mathbb{F}_q^2$, whose characters are folded as $(h(x,y), h(x,y'))$ and $(h_0(x), h_1(x))$, correspondingly, then the isomorphism preserves the Hamming distance of errors in these pairs: each character is independently acted upon by an invertible $\mathrm{GL}_2(\mathbb{F}_q)$ matrix.

The above protocol can thus be viewed as applying the isomorphism to replace $\langle h \wr \mathbf{S}_0 \rangle$ with $\langle (h_0, h_1) \wr \mathbf{T}_0 \rangle$, then moving to a random linear combination of h_0, h_1 to prove that both are proximal to W_0 simultaneously. Our main application for this protocol would involve proving proximity not for a single table $\langle h \wr \mathbf{S}_0 \rangle$ but multiple tables $\langle h_l \wr \mathbf{S}_0 \rangle$, $l = 1, \ldots, \mathsf{w}$ batched together. In this scenario we would prefer to first split each h_l into $h_{l,0}, h_{l,1}$ via our isomorphism, then take a random linear combination $\sum_{l=1}^{\mathsf{w}} \sum_{j=0}^{1} z_{l,j} h_{l,j}$ of all $2\mathsf{w}$ functions and apply FRI to it, batching all $2\mathsf{w}$ proofs together. An alternative, and perhaps more natural, approach would be to take first a random linear combination $\sum_{l=1}^{\mathsf{w}} z_l h_l$ of the w functions on $\mathbf{S}_0$, and then apply once the full proximity proof to U_0. The difference between the two approaches is subtle, and the main advantage of the first approach over the second is that its soundness analysis is again an immediate application of existing results.

7 Degree Adjustment

In Section 6.3 we discussed a proof protocol for showing an evaluation table $\langle f \wr \mathbf{S}_0 \rangle$ arises from a function $f \in \mathcal{K}_0 = \mathscr{L}([G_0])$. Our STARK proofs will need another, slightly more complex tool: we would like to show that certain functions also have zeros on a specified set of points; and on the other hand, the poles of these functions might not be simple and not at the entirety of G_0, but rather be contained in some subgroup $2^e G_0$, and have multiplicity at most 2^e each.

More explicitly, let $e \geq 1$ and let C be some coset of $2^{e+1}G_0$ which is disjoint from $\mathbf{S}_0$ and from G_0. Our goal in this section is to construct a proof of proximity to the AG code of evaluation tables $\langle f \wr \mathbf{S}_0 \rangle$ where $f \in \mathscr{L}(2^e[2^e G_0] - [I])$, where I is any non-empty subset of C; it will be assumed that f is already known to be in $\mathscr{L}(2^e[2^e G_0])$. This will be achieved by reducing this problem to the problem of proximity testing to U_0, where the reduction is performed by pointwise multiplication by a function ω_I with appropriate properties, listed below. For simplicity of presentation, we will henceforth assume $e = 1$; the results can be immediately generalized to larger e.

Definition 7.1. *A function $\omega_I \in k_{E_0}$ is called a* degree adjustment *for validity domain I, if it satisfies the following four properties:*

1. *For every $P \in 2G_0$, ω_I has (at least) a simple zero at P;*
2. *For every $P \in g_0 + 2G_0$, ω_I has (at most) a simple pole at P;*
3. *For every $P \in I$, ω_I has (exactly) a simple pole at P;*
4. *ω_I has no other poles except as specified above. It may have other zeros.*

The following claim shows that such a function enables the wanted reduction:

Lemma 7.2. *Let ω_I be a degree adjustment for a set $I \subset C$. Then for every $f \in \mathscr{L}(2[2G_0])$, we have*

$$f \text{ vanishes at all points in } I \Leftrightarrow \omega_I \cdot f \in \mathscr{L}([G_0]) = \mathcal{K}_0.$$

In other words, if it is known that a function f has poles only at $2G_0$ and of multiplicity at most 2, then f also vanishes at the points of I if and only if $\omega_I \cdot f$ is a $\mathcal{K}_0$ function, i.e. with at most simple poles and only at G_0.

Proof. In terms of divisors, properties 1–4 translate immediately to the inequality

$$\mathrm{div}(\omega_I) \geq [2G_0] - [g_0 + 2G_0] - [I] = (2[2G_0] - [I]) - [G_0],$$

from which the $\Rightarrow$ direction follows immediately by the definition of the spaces.

Note that the above inequality does not capture the conditions perfectly: specifically, property 3 states that ω_I *has* poles at the points of I, and rules out the possibility of the function being defined or vanishing at those points, which the inequality itself does not. This in turn is necessary (and sufficient) for the $\Leftarrow$ direction: if $\omega_I \cdot f \in \mathscr{L}([G_0])$, then it does not have poles at the points of I (which is disjoint from G_0). Since ω_I does have poles at those points, it follows that f must vanish at them. Since we already assume $f \in \mathscr{L}(2[2G_0])$, giving the necessary bounds on the location and multiplicity of poles, this is enough to deduce $f \in \mathscr{L}(2[2G_0] - [I])$. □

Our aim in this section is to construct such degree adjustments. Moreover, since these are to be used as part of a proof protocol, we need the construction to be well defined in terms of I, and succinct, so that both prover and verifier can compute it in reasonable time. We will show:

Theorem 7.3. *Let C be a coset of $2G_0$ disjoint from G_0. Denote $\overline{C} = G_0 \cup (C + G_0) \cup (-C + G_0)$, a union of three cosets of G_0. For every non-empty subset $I \subset C$, there exists a well-defined degree-adjustment ω_I. Moreover, for any point $P \in E_0 \setminus \overline{C}$, $\omega_I(P)$ is computable in $O(\|I\| + \mathsf{k})$ field operations, where $\|I\|$ is the coset complexity of I.*

In the following subsections we construct the degree adjustment in three steps. Each step will provide us with a function with some properties related to the definition of the degree adjustment. We denote these functions ν_0, μ_I, η_I (ν_0 will not depend on the zero-set I, only on the curve). The full degree adjustment will be the product of the three functions, $\omega_I = \nu_0 \cdot \mu_I \cdot \eta_I$.

A useful tool in the construction of these functions will be the following lemma:

Lemma 7.4. *Let $P, Q, R, S \in E$ be points on an elliptic curve with $P + Q = R + S$. Then there exists a function $r = r_{P,Q;R,S} \in k_E$ with*

$$\mathrm{div}(r) = [P] + [Q] - [R] - [S].$$

Moreover, $r(T)$ can be computed in $O(1)$ field operations for any $T \in E \setminus \{R, S, -R - S\}$.

Proof. Consider the equations defining the line passing through P, Q, and the line passing through R, S (if, say $P = Q$, then we use the tangent to the curve). Both lines pass through a common third point, $-P - Q = -R - S$. The ratio of the two equations gives a function as desired. Since the function is of the form $r(X, Y) = \frac{a_1 X + b_1 Y + c_1}{a_2 X + b_2 Y + c_2}$, clearly it is computable in $O(1)$ field operations whenever the denominator is non-zero. Additionally, extracting the coefficients of r from the points P, Q, R, S can also be done in $O(1)$ operations. □

Remark 7.5. Note that r is only well defined up to multiplication by a constant. For consistency we should normalize it such that the process always returns the same function. For example, we can normalize each line equation $aX + bY + c$ to have either $a = 1$, or $a = 0$ and $b = 1$, or $a = b = 0$ and $c = 1$.

The condition that $P + Q = R + S$ is necessary for such r to exist. In the next subsections, whenever we refer to a function $r_{P,Q;R,S}$, it will be implicit that $P + Q = R + S$ (and this will always be obvious to verify).

7.1 Step 1 - the independent part of the degree adjustment

In this section we construct the function ν_0, whose job is to shift the double poles at $2G_0$ to simple poles at G_0, i.e. to satisfy properties 1 and 2 of the degree adjustment. This is in fact too much to ask: due to structure of the curve, no function exists with exactly this property. We compromise by allowing ν_0 not to touch the pole at $\mathbf{0}$, and have a double zero at $\boldsymbol{\infty}$ instead.

More precisely, we construct functions $\nu_i \in k_{E_i}$ for $i \leq \mathsf{k} - 2$ with the following properties:

1. For every $P \in 2G_i \setminus \{\mathbf{0}, \boldsymbol{\infty}\}$, ν_i has a simple zero at P;
2. For every $P \in g_i + 2G_i$, ν_i has a simple pole at P;
3. ν_i has a *double zero* at $\boldsymbol{\infty}$;
4. ν_i has no other poles and zeros except as specified above, in particular it has no pole and no zero at $\mathbf{0}$.

In other words, ν_i must have the divisor

$$\mathrm{div}(\nu_i) = [2G_i] - [g_i + 2G_i] + [\boldsymbol{\infty}] - [\mathbf{0}],$$

which defines it up to a scalar. We thus show the following claim:

Claim 7.6. *There exists an explicit function $\nu_0 \in k_{E_0}$ with*

$$\mathrm{div}(\nu_0) = [2G_0] - [g_0 + 2G_0] + [\boldsymbol{\infty}] - [\mathbf{0}].$$

For any $P \notin G_0$, $\nu(P)$ can be computed using $O(\mathsf{k})$ field operations.

Proof. We construct ν_i by backwards induction, starting from $i = \mathsf{k} - 2$. We have that $2G_{\mathsf{k}-2} = \{\mathbf{0}, \boldsymbol{\infty}\}$ so we only need a double zero at $\boldsymbol{\infty}$ and simple poles at $\pm P_{\mathsf{k}-2}$, which is satisfied by the function

$$\nu_{\mathsf{k}-2}(X, Y) = \frac{1}{X - b_{\mathsf{k}-2}} = r_{\boldsymbol{\infty},\boldsymbol{\infty};P_{\mathsf{k}-2},-P_{\mathsf{k}-2}}(X, Y).$$

We then construct each ν_i from ν_{i+1} by

$$\nu_i(X,Y) = \frac{X - b_i}{X} \cdot \nu_{i+1}(\varphi_i(X,Y)).$$

The simple poles of ν_{i+1} at $g_{i+1}+2G_{i+1}$ are translated to simple poles of $\nu_{i+1}\circ\varphi_i$ at $g_i+2G_i = \varphi_i^{-1}(g_{i+1}+2G_{i+1})$. The simple zeros of ν_{i+1} at $2G_{i+1} \setminus \{\mathbf{0}, \boldsymbol{\infty}\}$ are translated to simple zeros of $\nu_{i+1}\circ\varphi_i$ at $2G_i \setminus \{\mathbf{0}, \boldsymbol{\infty}, \pm P_i\}$ and the double zero of ν_{i+1} at $\boldsymbol{\infty}$ is translated to double zeros of $\nu_{i+1}\circ\varphi_i$ at $\mathbf{0}$ and $\boldsymbol{\infty}$. The term $\frac{X-b_i}{X} = r_{P_i,-P_i;\mathbf{0},\mathbf{0}}(X,Y)$ then compensates for the lack of zero at $\pm P_i$ and the spare double zero at $\mathbf{0}$.

Using this recursive formula, the function ν_0 can be evaluated at any point outside of G_0 in just $O(\mathsf{k})$ field operations (which is logarithmic in the number of evaluation points). Moreover, when evaluating ν_0 on a coset of $2G_0$ we use $\nu_{i+1}(\varphi_i(P))$ both in the evaluation of $\nu_i(P)$ and of $\nu_i(P+\mathbf{0})$. Since the sizes of the layers shrink exponentially, it follows that evaluating ν_0 on an entire coset of $2G_0$ can be done in $O(|2G_0|)$ field operations. Moreover, this computation does not depend on I, so it can be done once and stored as a precomputation for any required ω_I. □

7.2 Step 2 - the part that depends on the validity domain

In this subsection we construct the function μ_I, which is the part of ω_I responsible for ensuring there are simple poles at the points of I (property 3). More precisely, we show the following:

Claim 7.7. *For any $I \subset C$, there exists an explicit function $\mu_I \in k_{E_0}$ with*

$$\mathrm{div}(\mu_I) = [|I|g_0] - [\boldsymbol{\infty}] + \sum_{P\in I}([P-g_0]-[P]).$$

For any $P \notin \overline{C}$, $\mu_I(P)$ can be computed in $O(\|I\|)$ field operations.

Note that such a function exists (and well defined up to a constant) since it has the same number of poles and zeros, and the sum of poles equals the sum of zeros. Additionally, since I is contained in a coset of $2G_0$ which is disjoint from G_0, all points in the divisor above are pairwise different, and in particular μ_I does have simple poles at every point of I.

Recall that the coset complexity of $\|I\|$ of I is the smallest possible sum $\sum_J(\log_2(|J|)+1)$ over all presentations $\mathbf{1}_I = \sum_J \epsilon_J \cdot \mathbf{1}_J$, where each J is a coset of a subgroup of $2G_0$. We first show how to construct each μ_J with the above divisor, then show that μ_J can be obtained from $\prod \mu_J^{\epsilon_J}$ and a small correction.

7.2.1 Periodic validity domain J

Since J is a coset of a subgroup of $2G_0$, we have $|J| = 2^m$ for some $m \le \mathsf{k}-1$. Denote $J_0 = J$ and define recursively $J_1,\ldots,J_m$ by $J_{i+1} = \varphi_i(J_i)$. We then have that each J_i is a coset of a subgroup $2G_i$ with $|J_i| = 2^{m-i}$. We construct $\mu_{J_i} \in k_{E_i}$ with

$$\mathrm{div}(\mu_{J_i}) = [2^{m-i}g_i] - [\boldsymbol{\infty}] + \sum_{P\in J_i}([P-g_i]-[P])$$

by backwards induction, starting from $i=m$.

For $i=m$ we have $J_m = \{P\}$. The function μ_{J_m} should satisfy

$$\mathrm{div}(\mu_{J_m}) = [g_m] + [P-g_m] - [\boldsymbol{\infty}] - [P],$$

so we take $\mu_{J_m} = r_{g_m,P-g_m;\boldsymbol{\infty},P}$, which is computable in $O(1)$ everywhere except $\pm P$ and $\boldsymbol{\infty}$.

Because $J_i = \varphi_i^{-1}(J_{i+1})$, we get

$$\mathrm{div}(\mu_{J_{i+1}} \circ \varphi_i) = [2^{m-i-1}g_i] + [2^{m-i-1}g_i + \mathbf{0}] - [\boldsymbol{\infty}] - [\mathbf{0}] + \sum_{P \in J_i} ([P - g_i] - [P]).$$

To get the right divisor we must therefore add a correction of

$$[2^{m-i}g_i] + [\mathbf{0}] - [2^{m-i-1}g_i] - [2^{m-i-1}g_i + \mathbf{0}],$$

which can do by defining

$$\mu_{J_i} = r_{2^{m-i}g_i, \mathbf{0}; 2^{m-i-1}g_i, 2^{m-i-1}g_i + \mathbf{0}} \cdot \mu_{J_{i+1}} \circ \varphi_i.$$

Putting this all together, we find that the computation of μ_J can be performed everywhere outside G_0 and $(J \cup -J) \subset (C \cup -C)$, in $O(m+1) = O(\log_2 |J| + 1)$ operations, and the computation of all relevant μ_J is done in $O(\|I\|)$ operations, and we conclude the proof.

7.2.2 Obtaining μ_I from $\prod_J \mu_J^{\epsilon_J}$

We now wish to obtain a way to compute μ_I from its components μ_J. To do this, it can be helpful to first expand the notion of μ_I from merely subsets $I \subset C$, to *signed multisubsets* of C. These are simply integer valued functions $I : C \to \mathbb{Z}$, and we would like the function μ_I to satisfy

$$\mathrm{div}(\mu_I) = [|I|g_0] - [\boldsymbol{\infty}] + \sum_{P \in C} I(P) \cdot ([P - g_0] - [P]),$$

where $|I| = \sum_{P \in C} I(P)$. Note that standard sets can be identified with their indicator functions, and the definition of $\mu_I = \mu_{\mathbf{1}_I}$ remains consistent. Since we have $\mathbf{1}_I = \sum_J \epsilon_J \cdot \mathbf{1}_J$, we see that to obtain $\mu_{\mathbf{1}_I}$ it will be sufficient to be able to relate $\mu_J, \mu_{J'}$ to $\mu_{J \pm J'}$, for any two signed multisets J, J', and apply this procedure repeatedly on the $\sum_J \epsilon_J J$. And indeed, it can be verified that

$$\mu_{J+J'} = r_{|J+J'|g_0, \boldsymbol{\infty}; |J|g_0, |J'|g_0} \cdot \mu_J \cdot \mu_{J'},$$
$$\mu_{J-J'} = r_{|J-J'|g_0, |J'|_0; |J|g_0, \boldsymbol{\infty}} \cdot \mu_J \cdot \mu_{J'}^{-1}$$

and both can be computed in $O(1)$ field operations, given the values of μ_J and $\mu_{J'}$. Thus the additional work of deriving μ_I from the $\{\mu_J\}$ is also bounded by $O(\|I\|)$ operations per evaluation.

7.3 Step 3 - the last small adjustments

We are almost done by now. The function $\nu_0 \cdot \mu_I$ almost matches all properties of ω_I. Its divisor is

$$\begin{aligned}
\mathrm{div}(\nu_0 \cdot \mu_I) &= \mathrm{div}(\nu_0) + \mathrm{div}(\mu_I) \\
&= [2G_0] - [g_0 + 2G_0] + [\boldsymbol{\infty}] - [\mathbf{0}] + [|I|g_0] - [\boldsymbol{\infty}] + \sum_{P \in I} ([P - g_0] - [P]) \\
&= ([2G_0] - [g_0 + 2G_0] - [I]) + [I - g_0] + [|I|g_0] - [\mathbf{0}],
\end{aligned}$$

and the only possible remaining issue is the "$-[\mathbf{0}]$" which appears at the end, meaning $\nu_0 \cdot \mu_I$ does not have a zero at $\mathbf{0}$ as it should (except in the special case $I = C$, where $|I|g_0 = \mathbf{0}$). On the other hand, we have extra zeros, at the points of $I - g_0$ and $|I|g_0$, which we may move to assist us in creating a zero at $\mathbf{0}$. We show the following:

Claim 7.8. *For any non-empty $I \subset C$, there exists an explicit function $\eta_I \in k_{E_0}$ such that*

$$\mathrm{div}(\eta_I) = [\mathbf{0}] - [|I|g_0] - [Q] + [Q'],$$

where $Q, Q' \in E_0$ are points such that $Q \in I - g_0$ and $Q' \notin I$. For any $P \notin \overline{C}$, $\eta_I(P)$ is computable in $O(1)$ field operations.

Proof. We first want to find a pair $Q \in I - g_0, Q' \notin I$ with $Q' = Q + |I|g_0 - \mathbf{0}$. This is equivalent to finding a point $P = Q + g_0 \in I$ such that $P + (|I| - 1 + 2^{\mathsf{k}-1})g_0 = Q' \notin I$. If no such pair exists, then since I is non-empty, it must be a union of cosets of $(|I| + 2^{\mathsf{k}-1} - 1)G_0$. These cosets are all even sized, so I must be even; but then it follows that these are cosets of G_0 itself, which is a contradiction since I is fully contained inside a coset of $2G_0$.

Having found such a pair, we have $Q + |I|g_0 = Q' + \mathbf{0}$, so we may simply define $\eta_I = r_{Q',\mathbf{0};Q,|I|g_0}$, which is computable in $O(1)$, and we are done. □

Finally, when we take $\omega_I = \nu_0 \cdot \mu_I \cdot \eta_I$, we obtain

$$\operatorname{div}(\omega_I) = ([2G_0] - [g_0 + 2G_0] - [I]) + ([I - g_0] - [Q]) + [Q']$$

which indeed satisfies all requirements, since the additional zeros at $(I - g_0)\backslash\{Q\} \cup \{Q'\}$ are all outside I. Furthermore, by checking the components we immediately see that $\omega_I(P)$ is computable for all $P \in E_0 \backslash \overline{C}$ in $O(\|I\| + \mathsf{k})$ operations, as claimed.

References

[AHIV17] Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *Proceedings of the 24th ACM Conference on Computer and Communications Security*, CCS '17, pages 2087–2104, 2017.

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. Preliminary version in FOCS '92.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. Preliminary version in FOCS '92.

[BBB+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *Proceedings of the 39th IEEE Symposium on Security and Privacy*, S&P '18, pages 315–334, 2018.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *ICALP*, volume 107 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[BBHR19] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable zero knowledge with no trusted setup. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO*, volume 11694 of *Lecture Notes in Computer Science*, pages 701–732. Springer, 2019.

[BCF+17] Eli Ben-Sasson, Alessandro Chiesa, Michael A. Forbes, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Zero knowledge protocols from succinct constraint detection. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, volume 10678 of *Lecture Notes in Computer Science*, pages 172–206. Springer, 2017.

[BCG+13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In *Proceedings of the 33rd Annual International Cryptology Conference*, CRYPTO '13, pages 90–108, 2013.

[BCG+17a] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Interactive oracle proofs with constant rate and query complexity. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 40:1–40:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[BCG+17b] Jonathan Bootle, Andrea Cerulli, Essam Ghadafi, Jens Groth, Mohammad Hajiabadi, and Sune K. Jakobsen. Linear-time zero-knowledge proofs for arithmetic circuit satisfiability. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 336–365. Springer, 2017.

[BCG+19] Eli Ben-Sasson, Alessandro Chiesa, Lior Goldberg, Tom Gur, Michael Riabzev, and Nicholas Spooner. Linear-size constant-query IOPs for delegating computation. In *Proceedings of the 17th Theory of Cryptography Conference*, TCC '19, 2019.

[BCG20] Jonathan Bootle, Alessandro Chiesa, and Jens Groth. Linear-time arguments with sublinear verification from tensor codes. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II*, volume 12551 of *Lecture Notes in Computer Science*, pages 19–46. Springer, 2020.

[BCGT13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 585–594. ACM, 2013.

[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.

[BCKL21] Eli Ben-Sasson, Dan Carmon, Swastik Kopparty, and David Levit. Elliptic curve fast fourier transform (ECFFT) part I: fast polynomial algorithms over all finite fields. *Electron. Colloquium Comput. Complex.*, page 103, 2021.

[BCR+19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 103–128. Springer, 2019.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In *Theory of Cryptography Conference*, pages 31–60. Springer, 2016.

[BFL91] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991. Preliminary version appeared in FOCS '90.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, STOC '91, pages 21–32, 1991.

[BGH+05] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Short PCPs verifiable in polylogarithmic time. In *Proceedings of the 20th Annual IEEE Conference on Computational Complexity*, CCC '05, pages 120–134, 2005.

[BGH19] Sean Bowe, Jack Grigg, and Daira Hopwood. Recursive proof composition without a trusted setup. Cryptology ePrint Archive, Report 2019/1021, 2019. https://ia.cr/2019/1021.

[BGKS20] Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. DEEP-FRI: sampling outside the box improves soundness. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 5:1–5:32. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[BKK+16] Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. Constant rate pcps for circuit-sat with sublinear query complexity. *J. ACM*, 63(4):32:1–32:57, 2016.

[BKS18] Eli Ben-Sasson, Swastik Kopparty, and Shubhangi Saraf. Worst-case to average case reductions for the distance to a code. In *Proceedings of the 33rd ACM Conference on Computer and Communications Security*, CCS '18, pages 24:1–24:23, 2018.

[BS08] Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM J. Comput*, 38(2):551–607, 2008.

[CC89] D. V. Chudnovsky and G. V. Chudnovsky. Computational problems in arithmetic of linear differential equations. some diophantine applications. In David V. Chudnovsky, Gregory V. Chudnovsky, Harvey Cohn, and Melvyn B. Nathanson, editors, *Number Theory*, pages 12–49, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg.

[CHM+20] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In *Proceedings of the 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '20, pages 738–768, 2020.

[CMS19] Alessandro Chiesa, Peter Manohar, and Nicholas Spooner. Succinct arguments in the quantum random oracle model. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 1–29. Springer, 2019.

[CMSZ21] Alessandro Chiesa, Fermi Ma, Nicholas Spooner, and Mark Zhandry. Post-quantum succinct arguments. *Electron. Colloquium Comput. Complex.*, page 38, 2021.

[COS20] Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 769–793. Springer, 2020.

[CY21a] Alessandro Chiesa and Eylon Yogev. Subquadratic snargs in the random oracle model. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part I*, volume 12825 of *Lecture Notes in Computer Science*, pages 711–741. Springer, 2021.

[CY21b] Alessandro Chiesa and Eylon Yogev. Tight security bounds for micali's snargs. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part I*, volume 13042 of *Lecture Notes in Computer Science*, pages 401–434. Springer, 2021.

[Deu41] Max Deuring. Die typen der multiplikatorenringe elliptischer funktionenkörper. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 14(1):197–272, Dec 1941.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In *Proceedings of the 32nd Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT '13, pages 626–645, 2013.

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for Muggles. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, STOC '08, pages 113–122, 2008.

[GPR21] Lior Goldberg, Shahar Papini, and Michael Riabzev. Cairo - a turing-complete stark-friendly CPU architecture. *IACR Cryptol. ePrint Arch.*, page 1063, 2021.

[Gro16] Jens Groth. On the size of pairing-based non-interactive arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 305–326. Springer, 2016.

[GS99] Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometry codes. *IEEE Trans. Information Theory*, 45(6):1757–1767, 1999.

[Gur07] Venkatesan Guruswami. Algorithmic results in list decoding. *Foundations and Trends® in Theoretical Computer Science*, 2(2):107–195, 2007.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019. https://ia.cr/2019/953.

[IKOS07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, page 21–30, New York, NY, USA, 2007. Association for Computing Machinery.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '92, page 723–732, New York, NY, USA, 1992. Association for Computing Machinery.

[Kob87] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48:203–209, 1987.

[Len87] H. W. Lenstra. Factoring integers with elliptic curves. *Annals of Mathematics*, 126(3):649–673, 1987.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992.

[MBKM19] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updateable structured reference strings. Cryptology ePrint Archive, Report 2019/099, 2019.

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, oct 2000.

[MMCRV05] Josep M. Miret, Ramiro Moreno Chiral, Anna Rio, and M. Valls. Determining the 2-sylow subgroup of an elliptic curve over a finite field. *Mathematics of Computation*, 74:411–427, 01 2005.

[PGHR13] Brian Parno, Craig Gentry, Jon Howell, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *Proceedings of the 34th IEEE Symposium on Security and Privacy*, Oakland '13, pages 238–252, 2013.

[RR21] Noga Ron-Zewi and Ron Rothblum. Proving as fast as computing: Succinct arguments with constant prover overhead. *Electron. Colloquium Comput. Complex.*, page 180, 2021.

[RRR16] Omer Reingold, Ron Rothblum, and Guy Rothblum. Constant-round interactive proofs for delegating computation. In *Proceedings of the 48th ACM Symposium on the Theory of Computing*, STOC '16, pages 49–62, 2016.

[Sha92] Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.

[Sil09] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*. Graduate texts in mathematics. Springer, Dordrecht, 2nd edition, 2009.

[Sta21] StarkWare. ethstark documentation. Cryptology ePrint Archive, Report 2021/582, 2021. https://eprint.iacr.org/2021/582.

[Was08] Lawrence C. Washington. *Elliptic Curves: Number Theory and Cryptography, Second Edition*. Chapman & Hall/CRC, 2 edition, 2008.

[Wat69] William C. Waterhouse. Abelian varieties over finite fields. *Annales scientifiques de l'École Normale Supérieure*, Ser. 4, 2(4):521–560, 1969.

A Mathematical background

In this section we survey some standard notions and notations from the theory of algebraic geometry and elliptic curves in particular. We list definitions and statements without proof; further details can be found in [Sil09, Was08].

A.1 Projective Space

We denote by $\mathbb{P}^n(\mathbb{F}_q)$ (or simply $\mathbb{P}^n$) the *n-dimensional projective space* over $\mathbb{F}_q$; only $\mathbb{P}^1$ and $\mathbb{P}^2$ will appear in the paper. Points in $\mathbb{P}^n$ are given by homogenized coordinates $[x_1 : x_2 : \cdots : x_{n+1}]$ where at least one x_i is non-zero, and with the equivalence relation

$$[x_1 : x_2 : \cdots : x_{n+1}] \sim [cx_1 : cx_2 : \cdots : cx_{n+1}], \quad \forall c \neq 0.$$

Points in the *affine* space $\mathbb{F}_q^n$ are given by affine coordinates $(x_1, \ldots, x_n)$, and in this paper we equate such points with their standard embedding into projective space, i.e.

$$(x_1, \ldots, x_n) = [x_1 : \cdots : x_n : 1].$$

Thus, $\mathbb{P}^n$ is the disjoint union of $\mathbb{F}_q^n$ and a copy of $\mathbb{P}^{n-1}$ "at infinity", i.e. with an additional $x_{n+1} = 0$ coordinate. In particular, $\mathbb{P}^1(\mathbb{F}_q) = \mathbb{F}_q \cup \{\infty\}$, where ∞ denotes the unique point at infinity, $[1 : 0]$.

We will refer to the two coordinates of the affine plane $\mathbb{F}_q^2$ as x and y. For a point $P \in \mathbb{F}_q^2$, we will denote its x, y coordinates by P_x, P_y, respectively. For a point $P \in \mathbb{P}^2$, the coordinates P_x, P_y will only be defined if it is an affine point, according to the above notation.

A.2 Elliptic Curves

An elliptic curve E over $\mathbb{F}_q$ is the set of solutions to a cubic equation in the projective plane, with the additional condition of being smooth. Every elliptic curve can be presented in extended Weierstrass form, where the defining equation is

$$E(X, Y): \quad Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6$$

with $a_1, a_2, a_3, a_4, a_6 \in \mathbb{F}_q$. All curves mentioned in this paper are assumed to be represented in Weierstrass form, and this will be essential when discussing the projection of the curve to $\mathbb{P}^1$. Every such curve passes through a unique

non-affine point, $\boldsymbol{\infty} = [0 : 1 : 0]$. The points of an elliptic curve form an abelian group, where $\boldsymbol{\infty}$ is the neutral element and for any three points $P, Q, R \in E$, $P + Q + R = \boldsymbol{\infty}$ if and only if the three points are colinear (and if any two are equal, the line must be tangent to the curve at that point). The rank of the group is at most 2, meaning it is of the form $\mathbb{Z}/m_1\mathbb{Z} \times \mathbb{Z}/m_2\mathbb{Z}$, where $m_2 \mid m_1$. The size of the group $m_1 \cdot m_2$ is in the Hasse–Weil bound $[q+1 \pm 2\sqrt{q}]$.

A.3 Rational functions

Rational functions over $\mathbb{F}_q$ are quotients $R(X) = P(X)/Q(X)$ where $P, Q \in \mathbb{F}_q[X]$ are coprime polynomials and Q is non-zero. Rational functions form a field, denoted by $\mathbb{F}_q(X)$, as well as $k_{\mathbb{P}^1}$.

Rational functions can be considered as maps from $\mathbb{P}^1$ to itself, where zeros of Q are mapped to ∞ and are called *poles* of the rational function, with multiplicity equal to their multiplicity as zeros of Q. Depending on whether $\deg(P) - \deg(Q)$ is positive, negative, or zero, the point ∞ is either a pole of multiplicity $\deg(P) - \deg(Q)$, a zero of multiplicity $\deg(Q) - \deg(P)$, or mapped to the ratio between the leading coefficients of P and Q, correspondingly.

The *degree* of R is defined as $\deg(R) \coloneqq \max(\deg(P), \deg(Q))$, and is equal to both the total number of zeros and the total number of poles of R, including at ∞, counted with multiplicity.

Rational functions over $\mathbb{F}_q^2$ are simply ratios of polynomial in $\mathbb{F}_q[X, Y]$. Given an elliptic curve E, the rational functions on E are simply restrictions of rational functions of $\mathbb{F}_q^2$ to E, and are thus defined only modulo the equation of E. The field of rational functions on E is denoted k_E, and is equal to the function field $\mathbb{F}_q(X, Y)/(E(X, Y))$. A non-zero rational function over E can also have zeros or poles at some points of E, with multiplicities. The total number of zeros will always equal the total number of poles, and be called the *degree* of the rational function.

A.4 Rational functions between varieties

A rational function between elliptic curves is a function $\varphi : E \to E'$ whose coordinates are given by rational functions. If a rational function also maps the neutral element of E to the neutral element of E', then it is also a group homomorphism, and is called an *isogeny*.

For any rational function $\varphi : A \to B$ between varieties (in our cases: $\mathbb{P}^1 \to \mathbb{P}^1$, $E \to \mathbb{P}^1$ or $E \to E'$), the *fibers* of φ are the preimages $\{\varphi^{-1}(P) : P \in B\}$. When considered over the algebraic closure $\overline{\mathbb{F}}_q$, almost all fibers are of the same maximal size. This size is the *separable degree* of φ. In this paper we shall deal only with separable functions, where the separable degree is also the degree itself (and this definition coincides with the previous definitions for functions to $\mathbb{P}^1$). The points $P \in B$ where $|\varphi^{-1}(P)| = \deg(\varphi)$ are called *unramified*, and the points where $|\varphi^{-1}(P)| < \deg(\varphi)$ are ramification points; over the algebraic closure, all but finitely many points are unramified. For an isogeny of curves, because it is a group homomorphism, it is unramified everywhere, and its degree is equal to the size of the kernel.

The map $\varphi : A \to B$ induces a *pullback* map $\varphi^* : k_B \to k_A : f \mapsto f \circ \varphi$. The zeros and poles of $\varphi^*(f) = f \circ \varphi$ are exactly at points P such that $\varphi(P)$ is a zero or pole of f, i.e., at the fibers of φ over the zeros and poles of f. For every $P \in A$, there is a *ramification index* $e_P \geq 1$ which satisfies that f has a zero/pole of multiplicity m at $\varphi(P)$ iff $f \circ \varphi$ has a zero/pole of multiplicity $e_P \cdot m$ at P, for every m. In each fiber, the sum of all ramification indices of the points is equal to the degree. Thus a point is unramified iff all ramification indices of points above it are 1, and zeros and poles of f translate to zeros and poles of f of the same multiplicity over the fiber. A special case of interest is that if φ has degree 2, then every point is either unramified, or its fiber is a single point with ramification index 2.

A.5 Divisors and Riemann–Roch spaces

For a variety A (either $\mathbb{P}^1$ or an elliptic curve for our purposes), its *divisor group* $\mathrm{Div}(A)$ is a free abelian group generated by the symbols $\{[P] : P \in A\}$, whose elements are called divisors. A divisor $D = \sum_{P \in A} n_P [P]$ is said to be non-negative if $n_P \geq 0$ for every $P \in A$. The degree of a divisor is $\deg(D) = \sum n_P$. Divisors have a partial order

given by $D \geq D'$ iff $D - D'$ is non-negative. For sets $S \subset A$, we abuse the notation and write $[S]$ as shorthand for the divisor $\sum_{P \in S}[P]$.

Let $k_A^\times = k_A \setminus \{0\}$ be the set of all non-zero rational functions on f. For any $f \in k_A^\times$, the divisor of f is $\mathrm{div}(f) = \sum m_P[P]$, where the sum is taken over all zeros and poles of f: for zeros m_P equals their multiplicity, and for poles m_P is minus the multiplicity. Thus $\deg(\mathrm{div}(f)) = 0$, since the total multiplicities of zeros and poles are equal. Divisors of the form $\mathrm{div}(f)$ are called *principal divisors*. For $\mathbb{P}^1$, every divisor of degree 0 is a principal divisor. For elliptic curves, a divisor $\sum n_P[P]$ of degree 0 is principal iff $\sum n_P P = \infty$ according to the group structure.

For a given divisor D, the *Riemann–Roch space* associated with D is

$$\mathscr{L}(D) = \{f \in k_A^\times : \mathrm{div}(f) \geq -D\} \cup \{0\}, \tag{30}$$

which is a linear space over $\mathbb{F}_q$. Its dimension is denoted by $\ell(D) = \dim \mathscr{L}(D)$. Note that since $\deg(\mathrm{div}(f)) = 0$, it follows that $\ell(D) = 0$ whenever $\deg(D) < 0$. The *Riemann–Roch theorem* is a useful tool for estimating $\ell(D)$ when $\deg(D) \geq 0$. We won't describe it in general, but mention its applications for the projective line and elliptic curves:

- In $\mathbb{P}^1$, for any divisor D with $\deg(D) \geq 0$, $\ell(D) = \deg(D) + 1$.
- In an elliptic curve, for any divisor D with $\deg(D) \geq 1$, $\ell(D) = \deg(D)$. If $\deg(D) = 0$, then $\ell(D) \in \{0, 1\}$, and the value depends on whether D is a principal divisor or not.

A.6 The Johnson Bound

We say that a code $V \subset \mathbb{F}_q^n$ is (γ, ℓ)-*list decodable* if for every $u \in \mathbb{F}_q^n$, there are no more than ℓ codewords of V that are within relative Hamming distance at most γ from u. Our first result is the Johnson bound for RS codes; see, e.g., [Gur07, Theorem 3.3] for a proof of this particular version.

Theorem A.1 (Johnson bound)**.** *For every $\eta \in (0, 1 - \sqrt{\rho})$, the code $\mathsf{RS}[\mathbb{F}_q, \mathbf{T}, \rho]$ is $(1 - \sqrt{\rho} - \eta, 1/(2\eta\sqrt{\rho}))$-list-decodable.*

B FRI over all finite fields

We begin by recapping (verbatim from [BGKS20]) the FRI protocol, as well as its main technical ingredient, a certain algebraic hash function based on the classical FFT (which hence constrains the ambient finite field to be special). We will then introduce a new (but closely related) algebraic hash function based on the ECFFT, that can be defined over all finite fields. Finally, we sketch the proof that original FRI protocol with the new algebraic hash function has the desired soundness properties over all finite fields — here we simply state the key features of the algebraic hash function that were needed for the soundness analysis from [BCI+20] of the original FRI protocol to work, and observe that the new algebraic has function also has them.

B.1 FRI

Our starting point is a function $f_0 : L_0 \to \mathbb{F}$ where $\mathbb{F}$ is a finite field, the evaluation domain $L_0 \subset \mathbb{F}$ is a coset of a group[13] contained in $\mathbb{F}$, and $|L_0| = 2^{k_0}$. We assume the target rate is $\rho = 2^{-\mathcal{R}}$ for some positive integer $\mathcal{R}$. The FRI protocol is a two-phase protocol (the two phases are called COMMIT and QUERY) that convinces a verifier that f_0 is close to the Reed–Solomon code $\mathsf{RS}[\mathbb{F}, L_0, \rho]$.

[13] The group can be additive, in which case $\mathbb{F}$ is a binary field, or multiplicative, in which case it is not.

The COMMIT phase of the FRI protocol involves $\mathsf{k} = k_0 - \mathcal{R}$ rounds. Before any communication, the prover and verifier agree on a sequence of (cosets of) sub-groups L_i, where $|L_i| = 2^{k_0 - i}$. Let RS_i denote the Reed–Solomon code $\mathsf{RS}[\mathbb{F}, L_i, d_i]$, where $d_i = \rho \cdot |L_i|$.

The main ingredient of the FRI protocol is a special algebraic hash function H_z, which takes a seed $z \in \mathbb{F}$, and given as input a function $f : L_i \to \mathbb{F}$, it produces as output a hash whose length is $1/2$ as long as f. More concretely, $H_z[f]$ is a function

$$H_z[f] : L_{i+1} \to \mathbb{F}$$

with the following properties:

1. **locality:** For any $s \in L_{i+1}$, $H_z[f](s)$ can be computed by querying f at just two points in its domain (these two points are $(q_i)^{-1}(s)$).
2. **completeness:** If $f \in \mathsf{RS}_i$, then for all $z \in \mathbb{F}$, we have that $H_z[f] \in \mathsf{RS}_{i+1}$.
3. **soundness:** If f is far from RS_i, then with high probability over the choice of seed z, $H_z[f]$ is quite far from RS_{i+1}.

These last two properties roughly show that for random z, H_z preserves distance to Reed–Solomon codes. The original H_z from the FRI protocol was based on the classical FFT decomposition [BKS18]. In the next subsection, we propose a different H_z, based on ECFFT and Equation (12) in particular, that allows FRI to be generalized to all fields.

The high-level idea of the FRI protocol can then be described as follows. First we are in the COMMIT phase of the protocol. The verifier picks a random $z_0 \in \mathbb{F}$ and asks the prover to write down the hash $H_{z_0}[f_0] : L^{(1)} \to \mathbb{F}$. By Properties 2 and 3 above, our original problem of estimating the distance of f_0 to RS_0 reduces to estimating the distance of $H_{z_0}[f_0]$ to $\mathsf{RS}^{(1)}$ (which is a problem of $1/2$ the size). This process is then repeated: the verifier picks a random $z^{(1)} \in \mathbb{F}$ and asks the prover to write down $H_{z^{(1)}}[H_{z_0}[f_0]]$, and so on. After k rounds of this, we are reduced to a constant sized problem which can be solved in a trivial manner. However, the verifier cannot blindly trust that the functions $f^{(1)}, \ldots$ that were written down by the prover truly are obtained by repeatedly hashing f_0. This has to be checked, and the verifier does this in the QUERY phase of the protocol, using Property 1 above.

We describe the phases of the protocol below.

COMMIT **Phase:**

1. For $i = 0$ to $\mathsf{k} - 1$:
   (a) The verifier picks uniformly random $z_i \in \mathbb{F}$ and sends it to the prover.
   (b) The prover writes down a function $f_{i+1} : L_{i+1} \to \mathbb{F}$. (In the case of an honest prover, $f_{i+1} = H_{z_i}[f_i]$.)
2. The prover writes down a value $C \in \mathbb{F}_q$. (In the case of an honest prover, $f^{(\mathsf{k})}$ is the constant function with value $= C$).

QUERY **Phase:** (executed by the Verifier)

1. Repeat ℓ times:
   (a) Pick $s_0 \in L_0$ uniformly at random.
   (b) For $i = 0$ to $\mathsf{k} - 1$:
      i. Define $s_{i+1} \in L_{i+1}$ by $s_{i+1} = q_i(s_i)$.
      ii. Compute $H_{z_i}[f_i](s_{i+1})$ by making 2 queries to f_i.
      iii. If $f_{i+1}(s_{i+1}) \neq H_{z_i}[f_i](s_{i+1})$, then REJECT.
   (c) If $f^{(\mathsf{k})}(s^{(\mathsf{k})}) \neq C$, then REJECT.
2. ACCEPT

B.2 The new algebraic hash function

We now describe our new algebraic hash function H_z.

The description of the hash function requires us to first fix a certain chain of isogenies of elliptic curves, as we do to define the ECFFT. For each $i \in [0, \mathsf{k}]$ we let $L_i \subseteq \mathbb{F}_q$ be the set T_i of size $2^{\mathsf{k}-i}$ from that theorem, and let $\psi_i \in \mathbb{F}_q(X)$ be the degree 2 rational function that maps L_i in a 2-to-1 manner to L_{i+1}. Let $\psi_i = u_i(X)/v_i(X)$, where $u_i(X), v_i(X) \in \mathbb{F}_q[X]$ polynomials with $\deg(u_i), \deg(v_i) \leq 2$.

Given $z \in \mathbb{F}$, $f : L_i \to \mathbb{F}$, the hash of f with seed z is defined to be the function $H_z[f] : L_{i+1} \to \mathbb{F}$ as follows. For $s \in L_{i+1}$, let $s_0, s_1 \in L_i$ be the two roots of $\psi_i(X) - s$. Let $P_{f,s}(Z) \in \mathbb{F}[Z]$ be the unique degree ≤ 1 polynomial satisfying

$$P_{f,s}(s_0) = \frac{f(s_0)}{(v_i(s_0))^{(d_i/2)-1}}, \tag{31}$$

$$P_{f,s}(s_1) = \frac{f(s_1)}{(v_i(s_1))^{(d_i/2)-1}}. \tag{32}$$

Then we define

$$H_z[f](s) = P_{f,s}(z). \tag{33}$$

Observe that $H_z[f](s)$ can be computed by querying f on the set $\{s_0, s_1\}$, and we denote this set by $S_i(s)$. Note also that the definition of $H_z[f]$ for $f \in L_i$ explicitly depends on the degree $d_i = \rho|L_i|$ — this is a difference from the definition of H_z in the original FRI protocol, where the definition of $H_z[f]$ did not depend on the ρ at all.

To understand H_z better, it is instructive to see what it does to RS_i. Let $f \in \mathsf{RS}_i$. The underlying polynomial $f(X)$ thus has degree less than $d = \rho|L_i|$. We may write $f(X)$ as:

$$f(X) = (f_0(\psi_i(X)) + X \cdot f_1(\psi_i(X))) \cdot v(X)^{(d/2)-1}, \tag{34}$$

where each $f_i(Y)$ has degree less than $d/2$.

Now take any $s \in L_{i+1}$, and let $S_i(s) = \{s_0, s_1\}$. Substituting $X = s_i$ into the above equation and using the fact that $\psi_i(s_i) = s$, we get:

$$\frac{f(s_i)}{v(s_i)^{(d/2)-1}} = f_0(s) + s_i \cdot f_1(s).$$

Comparing with Equations (31) and (32), we get that:

$$P_{f,s}(Z) = f_0(s) + Z \cdot f_1(s),$$

and thus for all $s \in L_{i+1}$, we have:

$$H_z[f](s) = f_0(s) + z f_1(s).$$

Thus $H_z[f]$ equals $\langle f_0 + z f_1 \wr L_{i+1} \rangle$, and is thus an element of RS_{i+1}.

B.3 Analysis of the new FRI protocol

The proof of soundness of the new FRI protocol is almost identical to the proof of soundness of the original FRI protocol from [BCI+20]. Indeed, the only properties of the algebraic hash function H_z that are used in that proof are:

- For every $f : L_i \to \mathbb{F}_q$, there are two functions $g, h : L_{i+1} \to \mathbb{F}_q$ such that:
  - $H_z[f] = g + z \cdot h$,

– Let $\mu : L_i \to [0,1]$ be a weight function. Let $\mu' : L_{i+1} \to [0,1]$ be defined by:

$$\mu'(s) = \mathbf{E}_{t \in S_i(s)} \mu(t).$$

Then the μ-weighted agreement of f with RS_i is completely determined by g and h by the following formula:

$$\mathsf{agree}_{\eta}(f, \mathsf{RS}_i) = \mathsf{agree}_{\eta'}(g \oplus h, \mathsf{RS}_{i+1} \oplus \mathsf{RS}_{i+1}),$$

where for two vectors $u, v \in \mathbb{F}_q^n$, we let $u \oplus v \in (\mathbb{F}_q^2)^n$ be the vector whose ith coordinate equals $(u_i, v_i) \in (\mathbb{F}_q^2)$.

B.4 Batched FRI

In this section, we prove Theorem 3.7 on Batched FRI. To simultaneously check that a collection of functions $g_1, \ldots, g_k : T \to \mathbb{F}_q$ are all low-degree ($\deg(g_i) < d_i$), they key is to take a random linear combination of the g_i with appropriate polynomial coefficients (in order to adjust their claimed degrees to all be the same) and run the FRI protocol on this. The reason this works is the proximity gaps phenomenon for Reed–Solomon codes [BCI+20].

Formally, the Batched FRI protocol is given below:

1. For each $i \in [k]$, define $c_i = \rho|T| - 1 - d_i$.
2. For each $i \in [k]$, pick random $(r_i, r_i') \in \mathbb{F}_q^2$.
3. Consider the function $f : T \to \mathbb{F}_q$ given by:

$$f(x) = \sum_{i=1}^{k} (r_i + r_i' x^{c_i}) g_i(x).$$

4. Run FRI on f with degree parameter $\rho|T|$ and repetition parameter t.

The analysis is identical to the analysis of Batched FRI in [BCI+20], which we sketch below. If f passes the FRI step with good probability, then f itself must be close to degree $< \rho|T|$. The key point is that f is a random linear combination of the functions g_i and the functions g_i', where $g_i'(x) = x^{c_i} g_i(x)$. By the basic Proximity Gaps result of [BCI+20], if f ends up being close to degree $< \rho|T|$ with good probability, then *all* the functions g_i and g_i' must themselves agree with degree polynomials G_i and G_i' on a large common agreement set. By the definition of g_i', this means that $G_i(X)$ and $X^{c_i} G_i(X)$ are both polynomials of degree $< \rho|T|$, and this means that for each i, $G_i(X)$ (which we know is close to g_i) is in fact a polynomial of degree $< \rho|T| - c_i$, as desired.