Scalable and Transparent Proofs over All Large Fields, via Elliptic Curves
(ECFFT Part II)

Eli Ben-Sasson[1][0000−0002−0708−0483], Dan Carmon[1][0000−0001−9952−5947], Swastik Kopparty[2][2222−−3333−4444−5555], and David Levit[1][0000−0001−6334−3738]

[1] StarkWare Industries Ltd., Netanya, Israel
https://starkware.co
{eli,dancar,david}@starkware.co
[2] University of Toronto, Toronto, Canada
swastik@cs.toronto.edu

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.

1 Introduction

Arithmetization was first used to construct interactive proofs in the seminal work of Lund et al. [41] and shortly after played a key role in Shamir's proof of

$\mathsf{IP} = \mathsf{PSPACE}$ [47]. 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[3], 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) $\mathsf{MIP} = \mathsf{NEXP}$ [5], (ii) the poly-logarithmic verification of NP [4] and (iii) the PCP Theorem [2,3], 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[4] 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 [7, Definition 3.3] for an exact definition. Scalable PCP systems for any language in NEXP were presented by [20,18,13], 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)[5]. In spite of their improved efficiency, scalable PCPs are not used in practice because the exponents

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

[4] 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.

[5] 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

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 [16,45] that generalizes both IP and PCP. From a computational complexity point of view, IOP = MIP = PCP = NEXP (see [16]). 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 [7]. The requirement that $\mathbb{F}$ be an FFT-friendly field remained.

To summarize, early IP, 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.

1.1 Main Results

The language most naturally suited for creating scalable IOPs is that of arithmetic intermediate representations (AIR) [7,49]. 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 1). An AIR witness is a tuple of functions $f_1, \ldots, f_w : \mathsf{D} \to \mathbb{F}$ (see Definition 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 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 [49], for domain specific languages like Winterfell, and for universal (Turing complete) virtual machines like Cairo [34]. 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 [7]. 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 polylog expressions are 1.

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

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.

Theorem 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*[6]*. 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 [15]; due to the concrete considerations mentioned above, as well as space limitations, we focus only on AIR.

Remark 1 (Zero Knowledge). The construction used in Theorem 1 can be augmented to achieve perfect zero knowledge, just like the FFT-friendly version of it (Theorem 3) can be augmented to an IOP with perfect zero knowledge [11]. We omit the addition of zero knowledge from this version due to space considerations.

Remark 2 (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 [28,27,26].

Fast IOPs of Proximity for Reed–Solomon and Elliptic Curve codes A major step, and bottleneck, in the construction of 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) [6] 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 proof of Theorem 1 we also extend the FRI protocol from [6], and its analysis from [8], to hold over all fields, provided $|\mathbb{F}| \geq \Omega(n^2)$.

Theorem 2 (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.*

[6] 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.

There exists a subset $\mathsf{D}' \subseteq \mathbb{F}, |\mathsf{D}'| = n$*, such that the family of RS codes of rate*[7] $\rho = 2^{-\mathcal{R}}$ *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*

$$(\max\{(1-\delta), \sqrt{\rho}\} - o(1))^t .$$

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) an IOPP for a more general family of codes – which comprises 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 2 to this pair. See the online version [10] for details.

Applications to concrete scalability We briefly argue why Theorem 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 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

[7] The rate parameter, defined as the ratio between a code's dimension and its block-length, can be picked to be any constant $\rho < 1$, and affects the soundness error and proximity parameters; see [8] for state of the art soundness bounds as a function of rate.

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.

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 typically 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 polylog 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 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 [20] to the linear Fast RS IOP of Proximity (IOPP) protocol of [6] (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 [48,50,40]. 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. A brief overview of some relevant standard facts and terms related to elliptic curves may be found in the online version [10].

Our starting point is our recent work [9], 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 [9] 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 the online version [10] for details.

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*[8] 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.

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 $\mathbb{P}^1$ is the Riemann–Roch

[8] 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 [9]. The cyclicity is essential for arithmetization.

space associated with the divisor $D = d \cdot [\infty]$ 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.

Key ingredients for the new IOPs, and the relationship to ECFFT Part I [9] 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 [9].

- **Super-efficient EC code encoding:** This essentially comes from [9]. 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 the online version [10] 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'.
- **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 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.
- **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.

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 [43], NIZKs and succinct NIZKs [33], SNARGs, SNARKs, STARKs, and more. Realizations in code include Pinocchio [44], C-SNARKs [14], PLONK [32], Halo [23], Fractal [28], Marlin [25], Ligero [1], Sonic [42], Bulletproofs [24] 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" [35] and "MPC in the head" [38] 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 [44]. Several blockchain systems, including Zcash, Filecoin and Tornado cash use the popular and efficient Groth16 ZK-SNARK [37]. 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[9]. Other constructions that rely on number-theoretic assumptions but which do not require knowledge of exponent assumptions, nor bilinear pairings (e.g., BulletProofs 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 [21,22,46,36] 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 pre-processing entity trusted by it. In particular, our main results (Theorems 1 and 2) 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 [9] (cf. [31] for an earlier work on using elliptic curves to compute an FFT-like transform, as well as the discussion in [9] of that paper). Indeed, the sequence of isogenies used here is adapted from that work, and the EC-FRI protocol 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 [9].

Algebraic Geometry codes and PCPs/IOPs A line of works used algebraic geometry codes to obtain PCPs and IOPs with extremely efficient proof

[9] 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.

length and query complexity over constant size fields [19,12]. 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.

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 6). Then we state and discuss our main theorem (Theorem 4). 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 [7], using the more recent formulation in [49, Section 5], generalizing it slightly by using an abstract cyclic group instead of a multiplicative group[10] 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 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.

[10] An AIR can also be defined using Hamiltonian paths in affine graphs, but restricting to cyclic groups suffices for NEXP-completeness, see [7].

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). The second, less trivial, component, is the specification of the domain on which different constraints must be enforced (Definition 3).

Definition 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.

Definition 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 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 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 6 (AIR Language/Relation). *The AIR relation* $\mathsf{R}_{\mathsf{AIR}}$ *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 $\mathsf{R}_{\mathsf{AIR}}$ *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 3 (Field size). The definition above requires $|\mathbb{F}| > (\mathsf{d}|\mathsf{H}_0|)^2$. When this is not the case one may embed $\mathbb{F}$ in a finite extension field $\mathbb{K}$ which 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 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 $\mathsf{L_{AIR}}$

To state our main result we assume familiarity with the definition of an IOP, and briefly recall its main parameters [16,45].

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* $\mathsf{time_P}$ and verifier complexity $\mathsf{time_V}$, 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 [7] that the sub-language of $\mathsf{L_{AIR}}$ restricted to FFT-friendly fields has a scalable and transparent IOP of knowledge. Formally, let

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

The main theorem of [7] is:

Theorem 3 (STIK for $\mathsf{L_{AIR,FFT}}$ – Prior state of art). *There is an IOP protocol for the language $\mathsf{L_{AIR,FFT}}$ 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 $(\mathsf{A}, \vec{\mathsf{f}}) \in \mathsf{R_{AIR}}$, 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 instances*

$$\mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{l}, \mathsf{Cset}),$$

  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 $(\mathsf{A}, \vec{\mathsf{f}}) \in \mathsf{R_{AIR}}$. In particular, if $\mathsf{A} \notin \mathsf{L_{AIR}}$ then, for any Prover strategy, the verifier will reject with probability at least $1 - 2^{-t}$.*

Remark 4 (Soundness and knowledge soundness). Often in the analysis of interactive proofs, the soundness error parameter is smaller than the knowledge 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 3. 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[11] on the curve E.

Theorem 4 (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 3 above.*

The complexity parameters of the theorem, along with completeness, are argued along the lines of the proof of Theorem 3 (see [49, Section 5]). The most delicate part is the soundness analysis (as is always the case with IOP systems). The proof appears in the online version [10].

EC-STARKs Assuming the existence of a family of collision resistant hash functions, and replacing the interactive oracles with Merkle commitment schemes a la [39], one obtains an interactive Scalable Transparent ARgument of Knowledge (STARK) as defined in [7]. Alternatively, working in the random oracle model and applying the BCS reduction [16], 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., [39,43,16,30,29]). We point out that STARKs based on FFT-friendly fields (Theorem 3) 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 will have practical applications in certain settings (as discussed in Section 1.1).

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

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 7 (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*
$$\Pr\left[\langle \mathsf{V}^v \leftrightarrow \mathsf{P}(v)\rangle = \mathsf{accept}\right] \leq \mathsf{err}(\Delta(v,V))$$

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 5 (FRI over smooth fields [6,8]). *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 $\mathsf{time}_{\mathsf{P}} = O(n)$ and proof length $\mathsf{l} < n$,*
- *logarithmic query complexity $\mathsf{q} = t \cdot \log(n) + O(1)$ and verification time $\mathsf{time}_{\mathsf{V}} = O(t \log n)$*
- *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 6 (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 5.*

2.4 Fast IOPs of Proximity for Elliptic Curve Codes

We generalize Theorem 6 to certain algebraic geometry codes, evaluations of functions in a low-degree Riemann–Roch space over FFT-friendly subgroups of elliptic curves. To define the specific codes recall the definition of Algebraic Geometry (or Goppa) codes.

Definition 8 (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 7 (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 . 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 5.*

3 Scalable IOPs for AIRs over any large field

In this section we prove our main theorem – Theorem 4, relying on certain claims that are proved in later sections.

3.1 The ECFFT Infrastructure

The proof of Theorem 4 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).

Due to space constraints, we briefly copy some the information from those sections so that we can describe our main IOP construction in the next section.

The EC backbone The backbone of all of the constructions in this paper is the chain of 2-isogenies whose existence was shown in [9, Theorem 4.9], which we quote here:

Theorem 8. *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 this work 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 8.

Theorem 9. *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 8, 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. See the online version [10] for details.

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.

Proposition 1 (Setup). *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 the online version [10] for an explicit description). This 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 $\mathcal{M}^{\langle \ell \rangle}$.

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 the online version [10]. The bases that appear in the theorems are defined there. Following the notation in [9], 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 10 (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 11 (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 $\langle g \wr T \rangle$).*

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 the online version [10].

Theorem 12 (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 [8], 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 the online version [10].

Theorem 13 (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 12 and 13 are some explicit small constants that are each at most* 10.

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. See the online version [10] for details.

Theorem 14 (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 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.
- 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 Proposition 1 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 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 $\mathsf{A} = (\mathbb{F}, \mathsf{w}, \mathsf{h}, \mathsf{d}, \mathsf{s}, \mathsf{H}_0, \mathsf{g}, \mathsf{l}, \mathsf{Cset})$ 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 [49][12], 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 [8], and (ii) the DEEP query and quotienting method [17], 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.

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 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 10); 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 $\mathbf{T}$.

[12] Some optimizations from [49], which are important for practical considerations and could also be done here, are omitted for clarity.

Prover sends $\langle \widehat{f}_{l,m} \wr \mathbf{T} \rangle$ 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 $\mathbf{T}$.
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 $\widehat{f^{\mathsf{r}}} \in \mathcal{K}^{\langle \mathsf{h}+1 \rangle}$ as 2 univariate polynomials:

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

where $\widehat{f_m^{\mathsf{r}}} \in \mathcal{M}_{\mathsf{h}+1}$ for $m \in \{0,1\}$.
The prover then evaluates both univariate polynomials at the points of $\mathbf{T}$.

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

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 $\widehat{f_m^{\mathsf{r}}}(\mathsf{q})$. Following the DEEP philosophy [17], we can then incorporate these claimed evaluations of $\widehat{f}_{l,m}$ and $\widehat{f_m^{\mathsf{r}}}$ 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

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

then supposedly $\alpha_{j,l} = \widehat{f}_l(\mathsf{q} + j \cdot g)$ and $\beta = \widehat{f^{\mathsf{r}}}(\mathsf{q})$.
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 13 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.

Due to space limitations of the conference version, full proofs are omitted. Full details appear in the online version [10].

References

1. Ames, S., Hazay, C., Ishai, Y., Venkitasubramaniam, M.: Ligero: Lightweight sublinear arguments without a trusted setup. In: Proceedings of the 24th ACM Conference on Computer and Communications Security. pp. 2087–2104. CCS '17 (2017)
2. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M.: Proof verification and the hardness of approximation problems. Journal of the ACM **45**(3), 501–555 (1998), preliminary version in FOCS '92.
3. Arora, S., Safra, S.: Probabilistic checking of proofs: a new characterization of NP. Journal of the ACM **45**(1), 70–122 (1998), preliminary version in FOCS '92.
4. Babai, L., Fortnow, L., Levin, L.A., Szegedy, M.: Checking computations in polylogarithmic time. In: Proceedings of the 23rd Annual ACM Symposium on Theory of Computing. pp. 21–32. STOC '91 (1991)
5. Babai, L., Fortnow, L., Lund, C.: Non-deterministic exponential time has two-prover interactive protocols. Computational Complexity **1**, 3–40 (1991), preliminary version appeared in FOCS '90.
6. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Fast reed-solomon interactive oracle proofs of proximity. In: Chatzigiannakis, I., Kaklamanis, C., Marx, D., Sannella, D. (eds.) ICALP. LIPIcs, vol. 107, pp. 14:1–14:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2018), http://www.dagstuhl.de/dagpub/978-3-95977-076-7
7. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable zero knowledge with no trusted setup. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO. Lecture Notes in Computer Science, vol. 11694, pp. 701–732. Springer (2019)
8. Ben-Sasson, E., Carmon, D., Ishai, Y., Kopparty, S., Saraf, S.: Proximity gaps for reed-solomon codes. In: Irani, S. (ed.) 61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020. pp. 900–909. IEEE (2020). https://doi.org/10.1109/FOCS46700.2020.00088
9. Ben-Sasson, E., Carmon, D., Kopparty, S., Levit, D.: Elliptic curve fast fourier transform (ECFFT) part I: fast polynomial algorithms over all finite fields. Electron. Colloquium Comput. Complex. p. 103 (2021), https://eccc.weizmann.ac.il/report/2021/103
10. Ben-Sasson, E., Carmon, D., Kopparty, S., Levit, D.: Scalable and Transparent Proofs over All Large Fields, via Elliptic Curves (ECFFT Part II) (2022), https://eccc.weizmann.ac.il/report/2022/110
11. Ben-Sasson, E., Chiesa, A., Forbes, M.A., Gabizon, A., Riabzev, M., Spooner, N.: Zero knowledge protocols from succinct constraint detection. In: Kalai, Y., Reyzin, L. (eds.) Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10678, pp. 172–206. Springer (2017). https://doi.org/10.1007/978-3-319-70503-3_6
12. Ben-Sasson, E., Chiesa, A., Gabizon, A., Riabzev, M., Spooner, N.: Interactive oracle proofs with constant rate and query complexity. In: Chatzigiannakis, I., Indyk, P., Kuhn, F., Muscholl, A. (eds.) 44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland. LIPIcs, vol. 80, pp. 40:1–40:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2017). https://doi.org/10.4230/LIPIcs.ICALP.2017.40
13. Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E.: On the concrete efficiency of probabilistically-checkable proofs. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) Symposium on Theory of Computing Conference, STOC'13, Palo Alto,

CA, USA, June 1-4, 2013. pp. 585–594. ACM (2013). https://doi.org/10.1145/2488608.2488681
14. Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E., Virza, M.: SNARKs for C: Verifying program executions succinctly and in zero knowledge. In: Proceedings of the 33rd Annual International Cryptology Conference. pp. 90–108. CRYPTO '13 (2013)
15. Ben-Sasson, E., Chiesa, A., Goldberg, L., Gur, T., Riabzev, M., Spooner, N.: Linear-size constant-query IOPs for delegating computation. In: Proceedings of the 17th Theory of Cryptography Conference. TCC '19 (2019)
16. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Theory of Cryptography Conference. pp. 31–60. Springer (2016)
17. Ben-Sasson, E., Goldberg, L., Kopparty, S., Saraf, S.: DEEP-FRI: sampling outside the box improves soundness. In: Vidick, T. (ed.) 11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA. LIPIcs, vol. 151, pp. 5:1–5:32. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). https://doi.org/10.4230/LIPIcs.ITCS.2020.5
18. Ben-Sasson, E., Goldreich, O., Harsha, P., Sudan, M., Vadhan, S.: Short PCPs verifiable in polylogarithmic time. In: Proceedings of the 20th Annual IEEE Conference on Computational Complexity. pp. 120–134. CCC '05 (2005)
19. Ben-Sasson, E., Kaplan, Y., Kopparty, S., Meir, O., Stichtenoth, H.: Constant rate pcps for circuit-sat with sublinear query complexity. J. ACM **63**(4), 32:1–32:57 (2016). https://doi.org/10.1145/2901294
20. Ben-Sasson, E., Sudan, M.: Short PCPs with polylog query complexity. SIAM J. Comput **38**(2), 551–607 (2008)
21. Bootle, J., Cerulli, A., Ghadafi, E., Groth, J., Hajiabadi, M., Jakobsen, S.K.: Linear-time zero-knowledge proofs for arithmetic circuit satisfiability. In: Takagi, T., Peyrin, T. (eds.) 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. Lecture Notes in Computer Science, vol. 10626, pp. 336–365. Springer (2017). https://doi.org/10.1007/978-3-319-70700-6_12
22. Bootle, J., Chiesa, A., Groth, J.: Linear-time arguments with sublinear verification from tensor codes. In: Pass, R., Pietrzak, K. (eds.) Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12551, pp. 19–46. Springer (2020). https://doi.org/10.1007/978-3-030-64378-2_2
23. Bowe, S., Grigg, J., Hopwood, D.: Recursive proof composition without a trusted setup. Cryptology ePrint Archive, Report 2019/1021 (2019), https://ia.cr/2019/1021
24. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: Proceedings of the 39th IEEE Symposium on Security and Privacy. pp. 315–334. S&P '18 (2018)
25. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N.: Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In: Proceedings of the 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 738–768. EUROCRYPT '20 (2020)
26. Chiesa, A., Ma, F., Spooner, N., Zhandry, M.: Post-quantum succinct arguments. Electron. Colloquium Comput. Complex. p. 38 (2021), https://eccc.weizmann.ac.il/report/2021/038

27. Chiesa, A., Manohar, P., Spooner, N.: Succinct arguments in the quantum random oracle model. In: Hofheinz, D., Rosen, A. (eds.) Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part II. Lecture Notes in Computer Science, vol. 11892, pp. 1–29. Springer (2019). https://doi.org/10.1007/978-3-030-36033-7_1
28. Chiesa, A., Ojha, D., Spooner, N.: Fractal: Post-quantum and transparent recursive proofs from holography. In: Canteaut, A., Ishai, Y. (eds.) 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. Lecture Notes in Computer Science, vol. 12105, pp. 769–793. Springer (2020). https://doi.org/10.1007/978-3-030-45721-1_27
29. Chiesa, A., Yogev, E.: Subquadratic snargs in the random oracle model. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12825, pp. 711–741. Springer (2021). https://doi.org/10.1007/978-3-030-84242-0_25
30. Chiesa, A., Yogev, E.: Tight security bounds for micali's snargs. In: Nissim, K., Waters, B. (eds.) Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13042, pp. 401–434. Springer (2021). https://doi.org/10.1007/978-3-030-90459-3_14
31. Chudnovsky, D.V., Chudnovsky, G.V.: Computational problems in arithmetic of linear differential equations. some diophantine applications. In: Chudnovsky, D.V., Chudnovsky, G.V., Cohn, H., Nathanson, M.B. (eds.) Number Theory. pp. 12–49. Springer Berlin Heidelberg, Berlin, Heidelberg (1989)
32. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953 (2019), https://ia.cr/2019/953
33. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct NIZKs without PCPs. In: Proceedings of the 32nd Annual International Conference on Theory and Application of Cryptographic Techniques. pp. 626–645. EUROCRYPT '13 (2013)
34. Goldberg, L., Papini, S., Riabzev, M.: Cairo - a turing-complete stark-friendly CPU architecture. IACR Cryptol. ePrint Arch. p. 1063 (2021), https://eprint.iacr.org/2021/1063
35. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: Interactive proofs for Muggles. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing. pp. 113–122. STOC '08 (2008)
36. Golovnev, A., Lee, J., Setty, S., Thaler, J., Wahby, R.S.: Brakedown: Linear-time and post-quantum snarks for r1cs. Cryptology ePrint Archive, Report 2021/1043 (2021), https://ia.cr/2021/1043
37. Groth, J.: On the size of pairing-based non-interactive arguments. In: Fischlin, M., Coron, J. (eds.) 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. Lecture Notes in Computer Science, vol. 9666, pp. 305–326. Springer (2016). https://doi.org/10.1007/978-3-662-49896-5_11
38. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from secure multiparty computation. In: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing. p. 21–30. STOC '07, Association for Comput-

ing Machinery, New York, NY, USA (2007). https://doi.org/10.1145/1250790.1250794
39. Kilian, J.: A note on efficient zero-knowledge proofs and arguments (extended abstract). In: Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing. p. 723–732. STOC '92, Association for Computing Machinery, New York, NY, USA (1992). https://doi.org/10.1145/129712.129782
40. Koblitz, N.: Elliptic curve cryptosystems. Mathematics of Computation **48**, 203–209 (1987)
41. Lund, C., Fortnow, L., Karloff, H.J., Nisan, N.: Algebraic methods for interactive proof systems. Journal of the ACM **39**(4), 859–868 (1992)
42. Maller, M., Bowe, S., Kohlweiss, M., Meiklejohn, S.: Sonic: Zero-knowledge SNARKs from linear-size universal and updateable structured reference strings. Cryptology ePrint Archive, Report 2019/099 (2019)
43. Micali, S.: Computationally sound proofs. SIAM J. Comput. **30**(4), 1253–1298 (oct 2000). https://doi.org/10.1137/S0097539795284959
44. Parno, B., Gentry, C., Howell, J., Raykova, M.: Pinocchio: Nearly practical verifiable computation. In: Proceedings of the 34th IEEE Symposium on Security and Privacy. pp. 238–252. Oakland '13 (2013)
45. Reingold, O., Rothblum, R., Rothblum, G.: Constant-round interactive proofs for delegating computation. In: Proceedings of the 48th ACM Symposium on the Theory of Computing. pp. 49–62. STOC '16 (2016)
46. Ron-Zewi, N., Rothblum, R.: Proving as fast as computing: Succinct arguments with constant prover overhead. Electron. Colloquium Comput. Complex. p. 180 (2021), https://eccc.weizmann.ac.il/report/2021/180
47. Shamir, A.: IP = PSPACE. Journal of the ACM **39**(4), 869–877 (1992)
48. Silverman, J.H.: The Arithmetic of Elliptic Curves. Graduate texts in mathematics, Springer, Dordrecht, 2nd edn. (2009). https://doi.org/10.1007/978-0-387-09494-6
49. StarkWare: ethstark documentation. Cryptology ePrint Archive, Report 2021/582 (2021), https://eprint.iacr.org/2021/582
50. Washington, L.C.: Elliptic Curves: Number Theory and Cryptography, Second Edition. Chapman & Hall/CRC, 2 edn. (2008)