MERCURY: A multilinear Polynomial Commitment Scheme with constant proof size and no prover FFTs

Liam Eagen

Alpen Labs

Ariel Gabizon

Aztec Labs

February 26, 2025

Abstract

We construct a polynomial commitment scheme for multilinear polynomials of size n where constructing an opening proof requires $O(n)$ field operations, and $2n + O(\sqrt{n})$ scalar multiplications. Moreover, the opening proof consists of a constant number of field elements. This is a significant improvement over previous works which would require either

1. $O(n \log n)$ field operations; or
2. $O(\log n)$ size opening proof.

The main technical component is a new method of verifiably folding a witness via univariate polynomial division. As opposed to previous methods, the proof size and prover time remain constant *regardless of the folding factor*.

1 Introduction

1.1 Our results

bl blabla

2 Overview of technique

Remark 2.1. *In this overview, we use some of the notation defined in Sections 3.1 and 3.2.*

Table 1: Comparison of pairing-based ml-PCS.

Scheme	**Proof size**	**Prover Work**	**Verifier Work**
univariate-based e.g. [PH23]	$O(1)$ $\mathbb{F}$	$O(n \log n)$ $\mathbb{F}$,$O(n)$ $\mathbb{G}$	$O(\log n)$ $\mathbb{F}$, $O(1)$ $\mathbb{G}$
gemini [gem]	$O(\log n)$ $\mathbb{F}$	$O(n)$ $\mathbb{F}$, $3n$ $\mathbb{G}$	$O(n^2)$ $\mathbb{F}$,$\mathbb{G}$ 1
zeromorph [zer]	$O(\log n)$ $\mathbb{F}$	$O(n)$ $\mathbb{F}$, $2.5n$ $\mathbb{G}$	$6n$ $\mathbb{G}$ 1, n $\mathbb{G}$ 2, $O(n \log^2 n)$ $\mathbb{F}$
$\mathfrak{MERCURY}$ [?]	$O(1)$ $\mathbb{F}$	$O(n)$ $\mathbb{F}$, $2n + O(\sqrt{n})$ $\mathbb{G}$	$13n$ $\mathbb{G}$ 1 n $\mathbb{G}$ 2, $O(n \log^2 n)$ $\mathbb{F}$

Our technique is best thought of as an improvement of the ml-PCS from gemini [gem]. Let's start by recalling how gemini works. gemini commits to a multilinear function as a *univariate* KZG commitment[KZG10]. Specifically, fix a vector $f \in \mathbb{F}^n$ describing the function's values on the boolean cube, and a structured reference string of elements $\left\{ \left[x^i\right]_1 \right\}_{i<n}$. gemini outputs $\mathsf{cm} = \sum_{i<n} f_i \left[x^i\right]_1$ as a commitment to the multilinear function $M(X_0, \ldots, X_{s-1}) = \sum_{i<n} \mathbf{eq}(i, u) f_i$ where $u \in \mathbb{F}^{\log n}$.

Now suppose prover $\mathbf{P}$ wants to convince verifier $\mathbf{V}$ that $M(u) = v$, for some $u = (u_0, \ldots, u_{s-1})$. $\mathbf{P}$ will send commitments $\mathsf{cm}_1, \ldots, \mathsf{cm}_s$ to the s incremental restrictions leading to evaluation at u. Namely, to $M_1 = M(u_0, X_1, \ldots, X_{s-1})$, $M_2 = M(u_0, u_1, X_2, \ldots, X_{s-1}), \ldots, M_s = M(u_0, \ldots, u_{s-1})$. Assuming $\mathbf{P}$ sent commitments to the correct functions, all that is needed is to check that cm_s is the commitment to the constant v. Of course, the technically interesting part is to prove the commitments *are* to the correct functions! For this purpose, gemini exploits a connection between M and its corresponding univariate $f(X)$: Write $f(X) = f_0(X^2) + X f_1(X^2)$, for $f_0(X), f_1(X)$ of degree $< n/2$. Let $f_{u_1}(X)$ be the univariate corresponding to M_1 defined above. Then, we have

$$f_{u_1}(X) = (1 - u_1) f_0(X) + u_1 f_1(X).$$

Additionally, we can evaluate f_0 and f_1 via f using the equations

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

Thus, we can perform consistency checks between each pair $\mathsf{cm}_{i-1}, \mathsf{cm}_i$ showing cm_i is indeed the commitment to the next correct restriction. Of course, we get $O(s) = O(\log n)$ proof length due to sending this incremental sequence of restriction commitments.

Here is a first idea on how to reduce proof length (without increasing prover time). Protocols based on univariate polynomials allow us to do multilinear evaluation in $O(n \log n)$ prover time with constant proof size (e.g. Section 5 of [PH23]) . Choose a parameter t and set $b = 2^t$. We can run *only the first t rounds* of gemini, reaching a restricted multilinear on $n - t$ variables. If $n' = n/(2^t) \leq n/\log n$, we can now apply a univariate protocol running in $O(n' \log n')$ time, running This still doesn't take us to overall constant proof size - as we need to use a super-constant t to reach such n'. (For us $t = \log n/2$ will be optimal, although $t \geq \log\log n$ suffices here.)

This raises the question - can we "skip" the intermediate gemini rounds and send *only* the commitment cm_t, and prove it is consistent with the original cm? Extrapolating the gemini strategy in the natural way, we get the answer - yes, but not with constant proof size: We can decompose f into b polynomials of degree $< n/b$: $f(X) = \sum_{0\leq i<b} X^i f_i(X^b)$. As in the $b = 2$ case, one can show the univariate $f'(X)$ corresponding to M_t will be a linear combination of the $\{f_i(X)\}$. Moreover, evaluating the f_i using f (for the consistency check) can be done. However, it requires b evaluations of f. Specifically, f_i at (r^b) can be evaluated using f at $\left\{r, r\omega, \ldots, r\omega^{t-1}\right\}$ for a primitive t'th root of unity ω.

Our central innovation is a different way to prove cm_t is correct *with* constant proof size. Take any $\alpha \in \mathbb{F}$, and let $g(X) = f(X) \bmod X^b - \alpha$. Then,

$$g(X) = \sum_{0\leq i<b} X^i f_i(\alpha)$$

The restricted polynomial we are interested in

$$f'(X) = \sum_{0\leq i<b} \mathbf{eq}(i, u_1) f_i(X)$$

Proving correctness of g can be done via a quotient. What remains is to connect g and f'. We can use KZG [KZG10] to get $f'(\alpha) = \sum_{0\leq i<b} \mathbf{eq}(i, u_1) f_i(X)$. Now we can get the same evaluation via evaluating g as a mulitlinear!

Comparison to sumcheck It is instructive to see what happens if we try to get a similar result via a modification of the sumcheck protocol [**?**]. Note first that indeed a multilinear evaluation can be seen as a sum over the the committed function multiplied by the $\mathbf{eq}$ function:

$$\hat{f}(u) = \sum_{b\in B_n} \mathbf{eq}(b, u)\hat{f}(b).$$

The classic sumcheck protocol, like gemini, works by $\log n$ reductions of the domain size by a factor of two; each round fixing one more variable of the summed function. In the above spirit, we could look at a modified sumcheck protocol, where the first variable ranges over a domain of super-constant size b. The first round univariate P_1 would thus have degree $2b$ in our case. We can send a commitment to it rather than its coefficients as usually done to maintain constant proof size. However, *computing* P_1 would require superlinear time $O(n \log b)$ - as we need to perform a b-size FFT for n/b values of the second variable appearing in the sum.

3 Preliminaries

3.1 Terminology and conventions

We work with integer parameter n that we'll assume throughout the paper is of the form $n = 2^{2t}$ for integer $t > 0$. We'll denote its square root by $b := 2^t = \sqrt{n}$. We index vectors starting from zero. For example, for $g \in \mathbb{F}^b$ we have $g = (g_0, \ldots, g_{b-1})$.

We associate vectors with univariate polynomials in the following natural way: Given $g \in \mathbb{F}^b$ we denote $g(X) := \sum_{0 \leq i < b} g_i X^i$.

We make the convention that integer ranges in sums begin at zero if not specified otherwise. Thus, we write $g(X) = \sum_{i<b} g_i X^i$.

We assume vectors of size n are indexed by two indices ranging over $\{0, \ldots, b-1\}$. Thus, for $f \in \mathbb{F}^n$, we have $f = (f_{0,0}, \ldots, f_{0,b-1}, \ldots, f_{b-1,0}, \ldots, f_{b-1,b-1})$. Accordingly, for $0 \leq i < b$, we denote by f_i the vector $(f_{i,0}, \ldots, f_{i,b-1})$.

In particular, for $f \in \mathbb{F}^n$ we have under these notations that

$$f(X) := \sum_{i<b} X^i f_i(X^b) = \sum_{i<b} \sum_{j<b} f_{i,j} X^{i+j \cdot b}$$

3.2 Multilinear polynomials

Let $n = 2^{2t}$. We define the well-known **eq** multilinear polynomial in $4t$ variables.

$$\mathbf{eq}(x, y) := \prod_{i=1}^{t} (x_i y_i + (1 - x_i)(1 - y_i))$$

We have for $x, y \in \{0,1\}^{2t}$, $\mathbf{eq}(x, y) = 1$ when $x = y$ and $\mathbf{eq}(x, y) = 0$ otherwise.

We use the convention that an integer $0 \leq i < n$ can be used as an input to **eq** by interpreting i as its binary representation. Namely, for $0 \leq i < n$, $u \in \mathbb{F}^t$, $\mathbf{eq}(i, u) := \mathbf{eq}(i_1, \ldots, i_t, u)$ where $i = \sum_{j \in [t]} i_j 2^{j-1}$.

For $a \in \mathbb{F}^n$, we define $\mathsf{ml}(f)$ to be the multilinear polynomial obtaining f's values on the boolean cube. Namely,

$$\mathsf{ml}(a)(X_1, \ldots, X_t) := \sum_{i<n} \mathbf{eq}(i, X_1, \ldots, X_t) \cdot a_i.$$

3.3 The algebraic group model

We introduce some terminology from [GWC19] to capture analysis in the Algebraic Group Model of Fuchsbauer, Kiltz and Loss[FKL18].

In our protocols, by an *algebraic adversary* $\mathcal{A}$ in an SRS-based protocol we mean a $\mathsf{poly}(\lambda)$-time algorithm which satisfies the following.

- For $i \in \{1, 2\}$, whenever $\mathcal{A}$ outputs an element $A \in \mathbb{G}_i$, it also outputs a vector v over $\mathbb{F}$ such that $A =< v, \mathsf{srs_i} >$.

First we say our srs *has degree* Q if all elements of $\mathsf{srs_i}$ are of the form $[f(x)]_i$ for $f \in \mathbb{F}_{<Q+1}[X]$ and uniform $x \in \mathbb{F}$. In the following discussion let us assume we are executing a protocol with a degree Q SRS, and denote by $f_{i,j}$ the corresponding polynomial for the j'th element of $\mathsf{srs_i}$.

Denote by a, b the vectors of $\mathbb{F}$-elements whose encodings in $\mathbb{G}1, \mathbb{G}2$ an algebraic adversary $\mathcal{A}$ outputs during a protocol execution; e.g., the j'th $\mathbb{G}1$ element output by $\mathcal{A}$ is $[a_j]_1$.

By a "real pairing check" we mean a check of the form

$$(a \cdot T_1) \cdot (T_2 \cdot b) = 0$$

for some matrices T_1, T_2 over $\mathbb{F}$. Note that such a check can indeed be done efficiently given the encoded elements and the pairing function $e : \mathbb{G}1 \times \mathbb{G}2 \to \mathbb{G}_t$.

Given such a "real pairing check", and the adversary $\mathcal{A}$ and protocol execution during which the elements were output, define the corresponding "ideal check" as follows. Since $\mathcal{A}$ is algebraic when he outputs $[a_j]_i$ he also outputs a vector v such that, from linearity, $a_j = \sum v_\ell f_{i,\ell}(x) = R_{i,j}(x)$ for $R_{i,j}(X) := \sum v_\ell f_{i,\ell}(X)$. Denote, for $i \in \{1,2\}$ the vector of polynomials $R_i = (R_{i,j})_j$. The corresponding ideal check, checks as a polynomial identity whether

$$(R_1 \cdot T_1) \cdot (T_2 \cdot R_2) \equiv 0$$

The following lemma is inspired by [FKL18]'s analysis of [Gro16], and tells us that for soundness analysis against algebraic adversaries it suffices to look at ideal checks. Before stating the lemma we define the Q-DLOG assumption similarly to [FKL18].

Definition 3.1. *Fix integer Q. The* Q-DLOG assumption for $(\mathbb{G}1, \mathbb{G}2)$ *states that given*

$$[1]_1, [x]_1, \ldots, \left[x^Q\right]_1, [1]_2, [x]_2, \ldots, \left[x^Q\right]_2$$

for uniformly chosen $x \in \mathbb{F}$, the probability of an efficient $\mathcal{A}$ outputting x is $\mathsf{negl}(\lambda)$.

Lemma 3.2. *Assume the Q-DLOG for $(\mathbb{G}1, \mathbb{G}2)$. Given an algebraic adversary $\mathcal{A}$ participating in a protocol with a degree Q SRS, the probability of any real pairing check passing is larger by at most an additive* $\mathsf{negl}(\lambda)$ *factor than the probability the corresponding ideal check holds.*

See [GWC19] for the proof.

3.4 Polynomial commitment schemes for multilinear polynomials

Definition 3.3. *Let $n = 2^t$. A multi-linear polynomial commitment scheme (ml-PCS) consists of*

- $\mathsf{gen}(n)$ - *a randomized algorithm that outputs an SRS* srs *0.*
- $\mathsf{com}(f, \mathsf{srs})$ - *that given a polynomial $f \in \mathbb{F}^n$ returns a commitment* cm *to f.*
- *A public coin protocol* $\mathsf{open}(\mathsf{cm}, n, u, v)$ *between parties* $\mathbf{P}$ *and* $\mathbf{V}$. $\mathbf{P}$ *is given $f \in \mathbb{F}^n$.* $\mathbf{P}$ *and* $\mathbf{V}$ *are both given integer n,* cm*- the purported commitment to f, $u \in \mathbb{F}^t$ and $v \in \mathbb{F}$ - the purported value* $\mathsf{ml}(f)(u)$.

such that

- ***Completeness:*** *Suppose that ,* $\mathsf{cm} = \mathsf{com}(f, \mathsf{srs})$. *Then if* open *is run correctly with values n,* cm*, $u, v =$* $\mathsf{ml}(f)(u)$*,* $\mathbf{V}$ *outputs $accept$ with probability one.*

- ***Knowledge soundness in the algebraic group model:*** *There exists an efficient E such that for any algebraic adversary $\mathcal{A}$ the probability of $\mathcal{A}$ winning the following game is* $\mathsf{negl}(\lambda)$ *over the randomness of $\mathcal{A}$ and* gen.

  1. *Given* srs, *$\mathcal{A}$ outputs* n, cm.
  2. *E, given access to the messages of $\mathcal{A}$ during the previous step, outputs $f \in \mathbb{F}^n$.*
  3. *$\mathcal{A}$ outputs $u \in \mathbb{F}^t$, $v \in \mathbb{F}$.*
  4. *$\mathcal{A}$ takes the part of $\mathbf{P}$ in the protocol* open *with inputs* n, cm, u, v.
  5. *$\mathcal{A}$ wins if*
     - $\mathbf{V}$ *outputs ~~accept~~ at the end of the protocol.*
     - $\mathsf{ml}(f)(u) \neq v$.

4 Components

In this section we go over known components (with some new optimizations), that will be used in our main protocol in the next section.

4.1 Inner products in $O(b \log b)$ time.

For polynomials $g_1(X) = \sum_{i=0}^{d_1} a_i X^i, g_2 = \sum_{i=0}^{d_2} b_i X^i$ in $\mathbb{F}[X]$. We define $< g_1, g_2 >$ to be $\sum_{i=0}^{d} a_i b_i$ where $d := \min\{d_1, d_2\}$. We present the following protocol to verify $< g_1, g_2 >$ as A convenient way to get inner products in monomial similar to [BCC+16, MBKM19].

Batching inner product checks If we wish to show that $< g_1, g_2 >= v_1$ and $< h_1, h_2 >= v_2$.

1. $\mathbf{V}$ chooses random $\gamma \in \mathbb{F}$
2. $\mathbf{P}$ sends $S \in \mathbb{F}[X]$ such that

$$g_1(X)g_2(1/X)+g_1(X)g_2(1/X)+\gamma(h_1(X)h_2(1/X)+h_1(1/X)h_2(X)) = 2(v_1+\gamma v_2)+X\cdot S(X)$$

4.2 Multi-liner evaluations as inner products of univariate polynomials.

For $u \in \{0,1\}^t$ define the polynomial $P_u = \sum_{i<b} \mathbf{eq}(i,u) X^i$. Thus, we have for $g \in \mathbb{F}_{<b}[X]$, $\mathsf{ml}(()g) =< g, P_u >$.

A verifier Inner products.

4.3 Degree checks

This is based on Thakur [Tha23].

4.4 Batching

Common practice to obtain an inner product of tw

Degree checks

5 Univariate division

Claim 5.1. *Fix integers $b > 0$ and let $n = b^2$. Fix $\alpha \in \mathbb{F}$, and $f(X) \in \mathbb{F}_{<n}[X]$. Let $f_0(X), \ldots, f_{b-1}(X) \in \mathbb{F}_{<b}[X]$ be such that $f(X) = \sum_{i<b} X^i f_i(X^b)$. Let $g(X) \in \mathbb{F}_{<b}[X], q(X) \in \mathbb{F}[X]$ be such that*

$$f(X) = (X^b - \alpha) \cdot q(X) + g(X).$$

Then,

1. *$g(X) = \sum_{i<b} X^i f_i(\alpha)$.*
2. *The coefficients of $q(X)$ can be computed in $O(n)$ $\mathbb{F}$-operations.*

Proof. To see the first item, note that reduction $\bmod X^b - \alpha$ corresponds to substituting α into X^b inside each $f_i(X^b)$ in the expression $\sum_{i<b} X^i f_i(X^b)$. We proceed to the computation of $q(X)$. We compute for each $0 \le i < b$, the coefficients of the quotient $q_i(X) \in \mathbb{F}[X]$ such that

$$f_i(X) = q_i(X)(X - \alpha) + f_i(\alpha).$$

Using Horner's method for division by the linear polynomial $X - \alpha$ this requires only n multiplications and additions in $\mathbb{F}$. Now, we have that

$$f(X) = \sum_{i<b} X^i f_i(X^b) = \sum_{i<b} X^i \left(q_i(X^b)(X^b - \alpha) + f_i(\alpha) \right) = q(X)(X^b - \alpha) + g(X),$$

for $q(X) := \sum_{i<b} X^i q_i(X^b)$. Thus, the coefficients of $q(X)$ are simply the interleaving of the coefficients of the $\{q_i(X)\}$. $\square$

6 Main Construction

$\mathfrak{MERCURY}$ is the tuple $(\mathsf{gen}, \mathsf{com}, \mathsf{open})$ described next.

$\underline{\mathsf{gen}(n)\text{:}}$ Choose random $x \in \mathbb{F}$ and outputs $\left\{ [1]_1, [x]_1, \ldots, \left[x^{n-1}\right]_1, [1]_2, [x]_2 \right\}$

$\underline{\mathsf{com}(n, f, \mathsf{srs})\text{:}}$ Output $\sum_{i<b} \sum_{j<b} f_{i,j} \cdot \left[x^{i \cdot b + j}\right]_1$.

$\underline{\mathsf{open}(n, \mathsf{cm}, u, v; f)\text{:}}$

1. Committing to partial sums:
   (a) **P** computes the polynomial to $h(X) := \sum_{i<b} \mathbf{eq}(i,u_1) f_i(X)$. Note that the coefficient of X^j in $h(X)$ is $\sum_{i<b} \mathbf{eq}(i,u_1) f_{i,j}$ - hence we think of it as a commitment to partial sums.
   (b) **P** computes and sends $\mathsf{h} := [h(x)]_1$.
2. Committing to "folded" polynomial g:
   (a) **V** sends random $\alpha \in \mathbb{F}$.
   (b) **P** computes polynomials $g(X) \in \mathbb{F}_{<b}[X]$ and $q(X) \in \mathbb{F}[X]$ such that
   $$f(X) = (X^b - \alpha) \cdot q(X) + g(X).$$
   (c) **P** computes and sends $\mathsf{q} := [q(x)]_1$ and $\mathsf{g} := [g(x)]_1$.
3. Sending proofs of correctness for h and the degree of g:
   (a) **V** sends a random batching challenge $\gamma \in \mathbb{F}$.
   (b) **P** computes and sends $\mathsf{s} = [S(x)]_1$ where $S(X) \in \mathbb{F}[X]$ is such that
   $$g(X)P_{u_1}(1/X) + g(1/X)P_{u_1}(X) + \gamma \cdot (h(X)P_{u_2}(1/X) + h(1/X)P_{u_2}(X))$$
   $$= h(\alpha) + \gamma \cdot v + X \cdot S(X) + (1/X)S(1/X).$$
   (c) **P** computes and sends $\mathsf{d} := [D(x)]_1$ where
   $$D(X) := X^{b-1} g(1/X).$$
4. KZG evaluations:
   (a) **V** sends a random evaluation challenge $\mathfrak{z} \in \mathbb{F}$.
   (b) **P** sends the values $f_\mathfrak{z} := f(\mathfrak{z}), q_\mathfrak{z} := q(\mathfrak{z}), g_\mathfrak{z} := g(\mathfrak{z}), \bar{g}_\mathfrak{z} := g(1/\mathfrak{z}), h_\mathfrak{z} := h(\mathfrak{z}), \bar{h}_\mathfrak{z} := h(1/\mathfrak{z}), h_\alpha := h(\alpha), s_\mathfrak{z} := s(\mathfrak{z}), \bar{s}_\mathfrak{z} := s(1/\mathfrak{z}), D_\mathfrak{z} := D(\mathfrak{z})$.
   (c) **V** sends a random KZG batching challenge $\eta \in \mathbb{F}$.
   (d) **P** computes and sends the KZG opening proof $\pi_\mathfrak{z}$ for the values $f_\mathfrak{z}$ and $q_\mathfrak{z}$. That is $\pi_\mathfrak{z} := [H(x)]_1$ for
   $$H(X) := \frac{f(X) - f(\mathfrak{z}) + \eta(q(X) - q(\mathfrak{z}))}{X - \mathfrak{z}}.$$
   (e) **P** computes and send a batched KZG opening proof π' for the rest of the values sent in step 4b, as described in Section 4 of [BDFG20].
   (f) **V** checks the proof $\pi_\mathfrak{z}$ as in [KZG10]:
   $$e(\mathsf{cm} - [f_\mathfrak{z}]_1 + \eta(\mathsf{q} - [q_\mathfrak{z}]_1), [1]_2) = e(\pi_\mathfrak{z}, [x]_2).$$

(g) **V** checks the opening proof π' as described in [BDFG20].

(h) **V** checks the equation

$$g_{\mathfrak{z}} P_{u_1}(1/\mathfrak{z}) + \bar{g}_{\mathfrak{z}} P_{u_1}(\mathfrak{z}) + \gamma(h_{\mathfrak{z}} P_{u_2}(1/\mathfrak{z}) + \bar{h}_{\mathfrak{z}} P_{u_2}(\mathfrak{z})) = h_\alpha + \gamma v + \mathfrak{z} s_{\mathfrak{z}} + (1/\mathfrak{z})\bar{s}_{\mathfrak{z}}.$$

(i) **V** checks the equation $D_{\mathfrak{z}} = \mathfrak{z}^{b-1}\bar{g}_{\mathfrak{z}}$.

(j) If one of the checks in steps 4f-4i fails **V** outputs $\mathscr{reject}$. Otherwise **V** outputs $\mathscr{accept}$.

Runtime of $\mathbf{P}$: Computing $q(X)$ in step 2b requires $O(n)$ operations by Claim 5.1. Computing q and $\pi_{\mathfrak{z}}$ requires two MSMs of size n. All other steps are on polynomials of size $O(b) = O(\sqrt{n})$.

Proving knowledge soundness: Let $\mathcal{A}$ be an efficient algebraic adversary participating in the Knowledge Soundness game from Definition 3.3. We show its probability of winning the game is $\mathsf{negl}(\lambda)$. Let $f \in \mathbb{F}^n$ be the vector sent by $\mathcal{A}$ in the third step of the game such that $\mathsf{cm} = [1]_1 f(x)$. As $\mathcal{A}$ is algebraic, when sending the commitments h,q,g,s,d,$\pi_{\mathfrak{z}}$,π' during protocol execution it also sends polynomials $h(X), q(X), g(X), S(X), D(X), H(X), Q(X) \in \mathbb{F}_{<n}[X]$ such that the former are their corresponding commitments. Let E be the event that **V** outputs $\mathscr{accept}$. Note that the event that $\mathcal{A}$ wins the knowledge soundness game is contained in E. E implies all pairing checks have passed. Let $A \subset E$ be the event that one of the corresponding ideal pairing checks as defined in Section 3.3 didn't pass. According to Lemma 3.2, $\mathsf{prob}(A) = \mathsf{negl}(\lambda)$.

References

[BCC+16] J. Bootle, A. Cerulli, P. Chaidos, J. Groth, and C. Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. pages 327–357, 2016.

[BDFG20] D. Boneh, J. Drake, B. Fisch, and A. Gabizon. Efficient polynomial commitment schemes for multiple points and polynomials. *IACR Cryptol. ePrint Arch.*, page 81, 2020.

[FKL18] G. Fuchsbauer, E. Kiltz, and J. Loss. The algebraic group model and its applications. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, pages 33–62, 2018.

[gem]

[Gro16] J. Groth. On the size of pairing-based non-interactive arguments. In *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, pages 305–326, 2016.

[GWC19] A. Gabizon, Z. J. Williamson, and O. Ciobotaru. PLONK: permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *IACR Cryptology ePrint Archive*, 2019:953, 2019.

[KZG10] A. Kate, G. M. Zaverucha, and I. Goldberg. Constant-size commitments to polynomials and their applications. pages 177–194, 2010.

[MBKM19] M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn. Sonic: Zero-knowledge snarks from linear-size universal and updateable structured reference strings. *IACR Cryptology ePrint Archive*, 2019:99, 2019.

[PH23] S. Papini and U. Haböck. Improving logarithmic derivative lookups using GKR. *IACR Cryptol. ePrint Arch.*, page 1284, 2023.

[Tha23] S. Thakur. A flexible snark via the monomial basis. *IACR Cryptol. ePrint Arch.*, page 788, 2023.

[zer]