On Extractability of the KZG Family of Polynomial Commitment Schemes

Juraj Belohorec[1,2], Pavel Dvořák[2], Charlotte Hoffmann[3], Pavel Hubáček[1,2], Kristýna Mašková[1,2], and Martin Pastyřík[2]

[1] Institute of Mathematics, Czech Academy of Sciences, Prague, Czech Republic
{belohorec,hubacek,maskova}@math.cas.cz
[2] Charles University, Faculty of Mathematics and Physics, Prague, Czech Republic
koblich@iuuk.mff.cuni.cz
[3] Institute of Science and Technology Austria, Klosterneuburg, Austria
charlotte.hoffmann@ista.ac.at

Abstract. We present a unifying framework for proving the knowledge-soundness of KZG-like polynomial commitment schemes, encompassing both univariate and multivariate variants. By conceptualizing the proof technique of Lipmaa, Parisella, and Siim for the univariate KZG scheme (EUROCRYPT 2024), we present tools and falsifiable hardness assumptions that permit black-box extraction of the multivariate KZG scheme. Central to our approach is the notion of a canonical Proof-of-Knowledge of a Polynomial (PoKoP) of a polynomial commitment scheme, which we use to capture the extractability notion required in constructions of practical zk-SNARKs. We further present an explicit polynomial decomposition lemma for multivariate polynomials, enabling a more direct analysis of interpolating extractors and bridging the gap between univariate and multivariate commitments. Our results provide the first standard-model proofs of extractability for the multivariate KZG scheme and many of its variants under falsifiable assumptions.

Keywords: KZG polynomial commitment · Extractability · Proof of Knowledge of a Polynomial · ARSDH · GARSDH.

Table of Contents

1 Introduction

zk-SNARKs (*Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge*) are cryptographic protocols that enable a prover to convince a verifier they know a possibly secret witness for a particular statement without revealing any details beyond the statement's validity. For example, the prover can demonstrate the correct output of a computation on private inputs without disclosing those inputs themselves. These arguments are both *succinct* and *non-interactive* because the proof consists of a single short message from the prover to the verifier, which can be verified quickly. The practical importance of zk-SNARKs has been recognized in privacy-preserving applications in decentralized systems like blockchain, where ensuring both the correctness of computations and the confidentiality of underlying data is critical.

Polynomial commitment schemes (PCSs) [KZG10], crucial building blocks for practical zk-SNARKs, enable efficient and succinct verification of polynomial evaluations without revealing the polynomial itself. Provers can commit to a polynomial and later prove, with minimal communication, that it evaluates to a certain value at a specific point. This reduces communication and computational overhead, essential for the scalability of zk-SNARKs, making them practical for real-world applications that require fast and low-bandwidth proofs. Central results on zk-SNARKs constructed polynomial commitment schemes and proved their security properties in various models and under various assumptions [MBKM19, CHM+20, GWC19, BDFG21]. Significant focus has been on the scheme presented by Kate, Zaverucha, and Goldberg [KZG10] and its extensions [PST13, ZGK+17a], as it offers the best communication complexity.

A zk-SNARK, being an *Argument of Knowledge*, guarantees that the prover actually holds a valid witness (e.g., a solution to a hard problem or a secret key) for the statement being proved. This *knowledge-soundness* property goes beyond computational soundness—where it is merely infeasible for a dishonest prover to convince the verifier of a *false statement*—by ensuring that no convincing proof can be constructed without truly possessing the witness. It is fundamental that the polynomial commitment used in zk-SNARKs also possesses the knowledge-soundness property, which ensures that if a prover can provide convincing evaluation proofs with respect to some commitment, then there exists a well-defined polynomial that the prover must know. This prevents dishonest provers from creating fake proofs without actually knowing the polynomial, ensuring the commitment corresponds to a legitimate polynomial.

The extractability of the KZG scheme. The KZG polynomial commitment and its variants were exploited in various constructions of communication-efficient zk-SNARKs. However, most proofs of knowledge-soundness for KZG [ZGK+17a, CHM+20, GWC19] are limited by the reliance on knowledge assumptions or idealized models, such as the Algebraic Group Model (AGM) [FKL18]. Knowledge assumptions, being non-falsifiable, limit our reasoning about security in real-world cryptographic settings [Nao03]. Similarly, while the AGM is useful for analyzing algebraic hardness assumptions, it presumes an idealized environment that does not fully capture practical cryptographic interactions.

Recently, Lipmaa, Parisella, and Siim [LPS24a, LPS24b] introduced the *Adaptive Rational Strong Diffie-Hellmann* (ARSDH) assumption in bilinear groups and demonstrated that it can be used to prove the knowledge-soundness of the univariate KZG polynomial commitment and its batched variant, which are important for practical zk-SNARKs such as PlonK [GWC19]. Thus, they gave the first proof of knowledge-soundness for the univariate KZG commitment scheme in the standard model under a new, yet falsifiable, cryptographic assumption. Being the first of its kind, their work suggests many important open problems. Most importantly, the adversary's task in the ARSDH security game closely mirrors the structure of the final verification check performed by the verifier on a KZG evaluation proof. This means that the assumption and the proof of knowledge-soundness are intricately linked to the specific operations and algebraic properties of the univariate KZG scheme. Consequently, if one attempts to generalize the proof of knowledge-soundness from Lipmaa et al. to other variants of the KZG scheme – such as, e.g., for multivariate polynomials [PST13, LXZ+24, LLZ+24] – the ARSDH assumption may no longer be applicable or sufficient. These variants, in particular, are currently not addressed by the techniques from Lipmaa et al.

1.1 Our Contribution

In this work, we establish knowledge-soundness for most known variants of the KZG polynomial commitment scheme. To this end, we generalize the proof technique introduced in [LPS24a] into a unifying framework enabling a rigorous proof of knowledge-soundness for any pairing-based polynomial commitment schemes with a "KZG-like" verification check.

PCS extractability via canonical proofs of knowledge of a polynomial. The common approach to defining knowledge-soundness for a polynomial commitment scheme in the literature is to require knowledge-soundness from the evaluation proof or the interactive opening protocol for evaluation at a specific point. However, this straightforward and seemingly sensible choice results in a formal definition that unnecessarily emphasizes the specific value at an arbitrary evaluation point. Indeed, *the canonical use of a polynomial commitment scheme that requires some notion of extractability is when transforming a Polynomial Interactive Oracle Proof (PIOP) into a SNARK.* However, there the verifier needs to evaluate the committed polynomial at a *random* evaluation point x. Moreover, the actual value z at x is of no significant importance to the verifier outside of the scope of the PIOP.

Motivated by this observation, we define a general notion of a *Proof of Knowledge of a Polynomial* for a PCS (PoKoP), which is an interactive protocol for the "commitment relation" corresponding to the PCS, i.e., a proof of knowledge of a polynomial f represented by the commitment C held by the verifier. Note that, as just stated, a PoKoP might not be useful in a compiler from PIOPs to SNARKs as it does not guarantee that the verifier learns an evaluation of the committed polynomial at any point – the "knowledge" of a committed polynomial could be, in principle, established by protocols that do not explicitly evaluate the polynomial. However, the canonical application of polynomial commitments in this context itself defines what we call the *canonical* PoKoP *for a PCS*: Given the commitment C, the verifier samples a uniform evaluation point x and sends it to the prover who sends back the value z at x together with an evaluation proof or, in the case of polynomial commitment schemes with an interactive evaluation proof, they proceed with the interactive evaluation proof. With this perspective, proving the extractability of a PCS simply means establishing that the above canonical PoKoP for the PCS is an argument of knowledge for its commitment relation.

By studying the canonical PoKoP of the multivariate KZG scheme, we manage to establish a clear statement about the extractability of the multivariate KZG scheme from [PST13]. Additionally, we can directly rely on various general results about proofs of knowledge such as general forking lemmata or the known relationships between witness-extended emulation and knowledge-soundness. Finally, we are able to easily transfer our techniques to many variants of the multivariate KZG scheme.

KZG PCS family via an explicit polynomial decomposition lemma. Both the univariate KZG scheme [KZG10] and its multivariate variant [PST13] are based on the following simple, yet surprisingly useful, proposition from algebraic geometry: a polynomial $g(X_1, \ldots, X_n)$ vanishes on $(x_1, \ldots, x_n) \in \mathbb{F}^n$ if and only if it admits a decomposition $g(X_1, \ldots, X_n) = \sum_{i=1}^{n} q_i(X_1, \ldots, X_n)(X_i - x_i)$ for some polynomials q_i. In the context of polynomial commitments, we use the proposition with the polynomial $g(X_1, \ldots, X_n) = f(X_1, \ldots, X_n) - z$ to test the claim that $f(x_1, \ldots, x_n) = z$. By the proposition,

$$f(X_1, \ldots, X_n) - z = \sum_{i=1}^{n} q_i(X_1, \ldots, X_n)(X_i - x_i) \tag{1}$$

for some polynomials q_i if and only if $f(x_1, \ldots, x_n) - z = 0$. In the univariate case, the multivariate test simplifies to checking whether $f(X) - z = q(X)(X - x)$, where $q(X) = (f(X) - z)/(X - x)$ is the quotient after dividing the polynomial $f(X) - z$ by $X - x$. By the proposition, the remainder in this polynomial division is zero if and only if $f(x) = z$. Kate, Zaverucha, and Goldberg [KZG10] showed that, in the setting of bilinear groups, the evaluation test via the existence of a suitable quotient polynomial q leads to an extremely efficient polynomial commitment scheme for univariate polynomials. Subsequently, Papamanthou, Shi, and Tamassia [PST13] showed that, also for multivariate polynomials, the quotient polynomials q_i in eq. (1) can

be computed efficiently via sequential polynomial division by the linear factors $(X_1 - x_1), \ldots, (X_n - x_n)$, which allowed them to extend the univariate construction from [KZG10] to an arbitrary number of variables.

We observe and prove that multivariate polynomials also admit *explicit* quotient decompositions analogous to the univariate case. We use the corresponding *explicit* multivariate polynomial decomposition lemma (Lemma 1) to show various structural properties of the multivariate KZG scheme from [PST13] that allow us to eventually prove its knowledge-soundness under a natural generalization of the ARSDH assumption from [LPS24a]. Since multivariate polynomial decomposition via quotient polynomials is at the core of any "KZG-like" polynomial commitment scheme, our explicit multivariate polynomial decomposition lemma could be of independent interest.

Interpolating extractors under the ARSDH family of assumptions. The extraction strategy for the univariate KZG scheme presented in [LPS24a] proceeds, quite naturally, via interpolation from accepting evaluation proofs for a random interpolation domain as follows: Given a commitment key ck and commitment C supposedly computed from a polynomial f of degree n, their extractor works in rounds, where, in each round, it samples (without replacement) a uniform evaluation point $x \leftarrow \mathbb{F}$ and asks the prover for the evaluation and evaluation proof for x. The extractor terminates after receiving $n+1$ accepting evaluation proofs. Then it simply outputs the interpolating polynomial $L(X)$ of degree n consistent with all the $n+1$ pairs of evaluation points and values with accepting evaluation proofs.

To establish black-box extractability of the KZG scheme, [LPS24a] showed that 1) the above extraction strategy runs in expected polynomial time and 2) under their new ARSDH assumption, the interpolating polynomial $L(X)$ is consistent with the commitment C except with a negligible probability. In more detail, [LPS24a] proved a "batching lemma" ensuring that any set of $n+1$ accepting KZG evaluation proofs defining an interpolating polynomial $L(X)$ can be combined into a single group identity, which seems computationally hard to satisfy with $L(X)$ that is not consistent with C. In other words, if we assume the hardness of satisfying such a group identity (formalized by [LPS24a] in ARSDH) then, by their batching lemma, we get the consistency of the interpolating extractor with the commitment, proving the knowledge-soundness of the univariate KZG scheme.

Our proof of knowledge-soundness of the *multivariate* KZG scheme is also established by analyzing its *interpolating extractor* that gathers enough accepting transcripts and then outputs the interpolating multivariate polynomial $L(X_1, \ldots, X_n)$. To prove the consistency of $L(X_1, \ldots, X_n)$ with the commitment, we establish two multivariate batching lemmata (Lemmas 2 and 5) for the multivariate KZG scheme. Similarly to the univariate batching lemma from [LPS24a] and ARSDH, our two batching lemmata motivate two new computational hardness assumptions that can be viewed as further generalizations of the Rational Strong Diffie-Hellmann assumption [GR19]. The first new assumption is a natural extension of ARSDH to the context of multivariate KZG. In fact, the ARSDH from [LPS24a] is a special case of our new assumption when restricted to a univariate scheme, and, thus, we simply call it ARSDH. The second assumption, which we call GARSDH, is a strengthening and generalization of ARSDH. Conceptually, since the weaker ARSDH assumption is less tailored to the interpolating extractor, it is harder to use for arguing the consistency of its output with the commitment. On the other hand, the stronger GARSDH assumption allows for a more direct argument while, arguably, being more tailored to the multivariate interpolating extractor.

To summarize, our results provide the first standard-model proofs of extractability for the multivariate KZG scheme and many of its variants under falsifiable assumptions.

1.2 Other Related Work

Many works build upon the multivariate KZG scheme from [PST13] or its zero-knowledge variant [ZGK+17b], e.g., [AM14, ZGK+17a, ZGK+18, Set20, CBBZ23, CGG+23, LXZ+24, LLZ+24, PP24, DMS24, Lib24, BMM+24, WHZ24, LZW+24]. The polynomial decomposition lemma from [PST13] is used, for example, in [ZGK+18, WZXP22, Lib24]. The Pianist [LXZ+24] and HyperPianist [LLZ+24] constructions rely on extractability of the multivariate KZG scheme from [PST13]. The authors point to [ZGK+17a] and claim that extractability was proven there. However, the authors of [ZGK+17a] proved that a *modification* of the multivariate KZG scheme is extractable under the *non-falsifiable* power knowledge of exponent assumption.

1.3 Organization of the Paper

In Section 2, we give an extensive overview of our techniques, illustrating them on the bivariate KZG scheme and then explaining how to extend them to the general multivariate scheme and its variants. In Section 3, we recall some basic notation and definitions. In Section 4, we formally define proofs of knowledge of a polynomial (PoKoPs), prove our explicit polynomial decomposition lemma, and use it to present the KZG family and its PoKoPs. In Section 5, we introduce the n-variate ARSDH and GARSDH assumptions and use them to prove the special soundness of the KZG PoKoPs. In Section 6, we recall the [ACK21] extraction strategy, use it to prove the knowledge-soundness of the KZG PoKoPs, and discuss the relationship between PoKoPs and the notion of black-box extractability for polynomial commitments from [LPS24a]. In Section 7, we discuss other variants of the multivariate KZG scheme that can be shown extractable using our techniques. In Section 8, we analyze the security of ARSDH and GARSDH in the algebraic and generic group models. In Section 9, we revisit the notion of computational uniqueness of proofs for the KZG family. In Section 10, we discuss the notions of evaluation binding achieved under the SBDH assumption by the basic and randomized multivariate KZG schemes from [PST13].

2 Overview of Our Techniques

As we discussed in Section 1.1, when proving the knowledge-soundness of a polynomial commitment scheme in the standard model, it is natural to use the *interpolating extractor* that gathers accepting evaluation proofs for some feasible interpolation domain and then outputs the interpolating polynomial consistent with the received evaluations. The crux of the proof then lies in establishing that the interpolating polynomial is consistent with the received commitment C, which is the focus of the next two subsections.

Bivariate KZG. For concreteness, we illustrate our proof on the bivariate variant of the KZG scheme. Recall that, we are in a setting of bilinear groups, i.e., both prover and verifier have access to additive groups $\mathbb{G}_1, \mathbb{G}_2$ and a multiplicative "target" group $\mathbb{G}_T$, all of prime order p. Additionally, there is an efficiently computable *pairing*, i.e., a non-degenerate bilinear map $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. To simplify the notation and make algebraic manipulations clearer, we use the bracket notation for group elements, i.e., we write $[a]_1 = a[1]_1$, where $[1]_1$ is the generator of $\mathbb{G}_1$ (and similarly for $\mathbb{G}_2$). We use the symbol $\bullet$ as a binary operator that represents an application of the pairing, i.e., $[a]_1 \bullet [b]_2 = e([a]_1, [b]_2)$, and the symbol $\otimes$ for multiplication in the target group. Finally, we use $[n]$ to denote the set $\{0, \ldots, n\}$.

In the bivariate KZG scheme, the prover's input $f(X,Y) = \sum_{i=0}^{m}\sum_{j=0}^{n} f_{i,j}X^iY^j \in \mathbb{F}_p[X,Y]$ is a bivariate polynomial of maximal X-degree m and Y-degree n. As is common in any variant of the KZG scheme, the commitment C to $f(X,Y)$ is a discrete logarithm commitment $[f(\sigma,\tau)]_1$ to the value of f at some secret evaluation point $(\sigma,\tau) \leftarrow \mathbb{F}_p^2$. Importantly, the prover does not know (σ,τ) and, to compute the commitment $C = [f(\sigma,\tau)]_1$, receives a commitment key $\mathsf{ck} = \big(\{[1]_1, [1]_2, ([\sigma^i\tau^j]_1)_{i\in[m],j\in[n]}, [\sigma]_2, [\tau]_2, \big)$ comprising the commitments to the evaluations of all the standard basis polynomials, i.e., all the monomials of the form X^iY^j. Note that, given ck and the coefficients of the polynomial f, the commitment can be computed simply as $C = [f(\sigma,\tau)]_1 = \sum_{i=0}^{m}\sum_{j=0}^{n} f_{i,j}[\sigma^i\tau^j]_1$.

To compute the evaluation proof for value $z \in \mathbb{F}_p$ at $(x,y) \in \mathbb{F}_p^2$, the prover and verifier rely on the following identity:

$$f(X,Y) - f(x,y) = \frac{f(X,Y) - f(x,Y)}{X - x}(X - x) + \frac{f(x,Y) - f(x,y)}{Y - y}(Y - y),$$

where $q_1(X,Y) = \frac{f(X,Y)-f(x,Y)}{X-x}$ and $q_2(Y) = \frac{f(x,Y)-f(x,y)}{Y-y}$ are, in fact, polynomials. The correctness of the above decomposition is a special case of our explicit polynomial decomposition lemma (Lemma 1) we discussed in Section 1.1, when restricted to bivariate polynomials. Thus, the verifier tests the claim that $f(x,y) = z$ by verifying the correctness of the polynomial decomposition on the secret evaluation point (σ,τ), i.e., that

$$f(\sigma,\tau) - z = q_1(\sigma,\tau)(\sigma - x) + q_2(\tau)(\tau - y).$$

Even though neither the prover nor the verifier knows (σ, τ), the above check can be made efficient by exploiting the bilinearity of the pairing. The prover sends a proof $\pi = (\pi_1, \pi_2) = ([q_1(\sigma, \tau)]_1, [q_2(\tau)]_1)$ to the verifier, who, instead of the above identity over $\mathbb{F}$, checks the target group identity

$$[c - z]_1 \bullet [1]_2 = \pi_1 \bullet [\sigma - x]_2 \otimes \pi_2 \bullet [\tau - y]_2,$$

which is easy to compute given the commitment key ck, commitment $C = [c]_1$, evaluation point (x, y), claimed value z, and evaluation proof π.

2.1 Special Soundness via a Batching Lemma for Structured Accepting Evaluation Proofs

We wish to argue that, unless it breaks some hard computational problem, any prover who succeeds in constructing accepting evaluation proofs for $C = [c]_1$ at many evaluation points (x, y) in the bivariate KZG scheme is soon bound to the interpolating polynomial defined by the provided z-values. Specifically, it should be possible to construct a polynomial $L(X, Y)$ consistent with all the accepting evaluation proofs as well as the commitment $[c]_1$, i.e., such that $[c]_1 = [L(\sigma, \tau)]_1$ or, equivalently, $[c - L(\sigma, \tau)]_1 = [0]_1$.

First, we recall some useful notation for Lagrange interpolation (see Definition 1 for the formal definitions). For the purpose of this technical overview, $\{\ell_j^S(Y)\}_{j\in[n]}$ is simply the set of $n + 1$ polynomials of degree n such that, if $S = \{y_j\}_{j\in[n]}$ then for all $j \in [n]$, $\ell_j^S(y_j) = 1$ and $\ell_j^S(y) = 0$ for all $y \in S \setminus \{y_j\}$. Later, we use the fact that the polynomial $\sum_{j\in[n]} \ell_j^S(Y)$ is the constant identically 1 polynomial, which follows since it is a polynomial of degree at most n that evaluates to 1 on all of S.

Now, we discuss the construction of the interpolating polynomial $L(X, Y)$ from accepting transcripts in two steps. Suppose, we have accepting proofs for values $z_0, \ldots, z_n$ at distinct points $(x, y_0), \ldots, (x, y_n)$ that share the first coordinate. Then, we can take the corresponding univariate interpolating polynomial $L_x(Y)$ of degree n that agrees with the values $\{z_j\}_{j\in[n]}$ at the evaluation domain $\{(x, y_j)\}_{j\in[n]}$. Formally, for the univariate Lagrange basis $\{\ell_j^S(Y)\}_{j\in[n]}$ w.r.t. the evaluation domain $S = \{y_j\}_{j\in[n]}$, we define $L_x(Y) = \sum_{j=0}^n z_j \ell_j^S(Y)$. Note that if we were interacting with an honest prover then $L_x(Y)$ would agree with the univariate restriction $f_x(Y) = f(x, Y)$ of the prover's bivariate polynomial $f(X, Y)$. Even though we cannot assume that the prover is honest, under a suitable computational hardness assumption, we establish some strong consistency property of $L_x(Y)$ w.r.t. C for arbitrary computationally bounded prover.

Bivariate batching lemma. Our initial observation is that if, for each pair $((x, y_j), z_j)$, we had an accepting evaluation proof $\pi_j = ([\varphi]_1, [\psi_j]_1)$, i.e., with a shared first component $[\varphi]_1$,[4] satisfying the verification check

$$[c - z_j]_1 \bullet [1]_2 = [\varphi]_1 \bullet [\sigma - x]_2 \otimes [\psi_j]_1 \bullet [\tau - y_j]_2, \tag{2}$$

then we can establish that

$$[c - L_x(\tau)]_1 \bullet [1]_2 = [\varphi]_1 \bullet [\sigma - x]_2 \otimes [\psi]_1 \bullet [Z_S(\tau)]_2, \tag{3}$$

where $Z_S(Y) = \prod_{j=0}^n (Y - y_j)$ is the vanishing polynomial of the set $S = \{y_j\}_{j=0}^n$ and $[\psi]_1 = \sum_{j=0}^n w_j^S [\psi_j]_1$ with $w_j^S = \ell_j^S(Y)(Y - y_j)/Z_S(Y)$ for all $j \in [n]$.

The above transformation of the $n+1$ equalities in eq. (2) into eq. (3) involving the partial interpolating polynomial $L_x(Y)$ is a bivariate analogue of the univariate batching lemma from [LPS24a]. Since our proof of this bivariate batching lemma is very direct, we present it next. We expand and manipulate the left side

[4] If we interacted with an honest prover then the evaluation proofs would have shared π_1 for any pair of evaluation points that share the X-value since, by our explicit polynomial decomposition lemma, the first quotient polynomial q_1 does not depend on the Y-value. We address how to extract such structured transcripts from an arbitrary prover in the following section.

in eq. (3) to derive the right side:

$$\begin{aligned}
[c - L_x(\tau)]_1 \bullet [1]_2 &= \left[c - \sum_{j=0}^{n} z_j \ell_j^S(\tau)\right]_1 \bullet [1]_2 \\
&= \left[\sum_{j=0}^{n} \ell_j^S(\tau)(c - z_j)\right]_1 \bullet [1]_2 \\
&= \left[\sum_{j=0}^{n} \ell_j^S(\tau)\left(\varphi(\sigma - x) + \psi_j(\tau - y_j)\right)\right]_1 \bullet [1]_2 \\
&= \left[\sum_{j=0}^{n} \ell_j^S(\tau)\varphi(\sigma - x) + \sum_{j=0}^{n} Z_S(\tau) w_j^S \psi_j\right]_1 \bullet [1]_2 \\
&= [\varphi(\sigma - x) + Z_S(\tau)\psi]_1 \bullet [1]_2 \\
&= [\varphi]_1 \bullet [\sigma - x]_2 \otimes [\psi]_1 \bullet [Z_S(\tau)]_2.
\end{aligned}$$

The correctness of the above derivations can be justified as follows. In the first equality, we used the definition of $L_x(Y)$ via Lagrange interpolation. In the second equality, we used that $\sum_{j=0}^{n} \ell_j^S(Y) = 1$ (as we mentioned when introducing a Lagrange basis). In the third equality, we used eq. (2), and the bilinearity of the pairing. In the fourth equality, we used that $\ell_j^S(Y)(Y - y_j) = Z_S(Y) w_j^S$. In the fifth equality, we again used the fact that $\sum_{j=0}^{n} \ell_j^S(Y) = 1$, along with the definition of $[\psi]_1$. Finally, we just moved terms around using the bilinearity of the pairing.

Inductive proof using partial interpolants from the bivariate batching lemma. At first, eq. (3) might not seem like much progress. However, if it additionally held that $[c - L_x(\tau)]_1 \bullet [1]_2 = [\varphi]_1 \bullet [\sigma - x]_2$ then we would have obtained an accepting *univariate* KZG proof establishing that $f_\tau(X) = f(X, \tau)$ evaluates to $L_x(\tau)$ at x. Intuitively, we got one step closer to constructing a bivariate $L(X, Y)$ such that $[c - L(\sigma, \tau)]_1 = [0]_1$ as we now have a univariate $L_x(Y)$ that is consistent with C at (x, τ). If we had $m + 1$ such identities for distinct $x_0, \ldots, x_m$, we could combine the corresponding univariate polynomials $L_{x_i}(Y)$ via the Lagrange basis $\{\ell_i^U(X)\}_{i \in [m]}$ w.r.t. $U = \{x_i\}_{i \in [m]}$. By its construction, the resulting bivariate interpolating polynomial $L(X, Y) = \sum_{i=0}^{m} \ell_i^U(X) L_{x_i}(Y)$ would be a suitable candidate polynomial consistent with all the observed z values that our extractor could output. Importantly, to show that $[c - L(\sigma, \tau)]_1 = [0]_1$, i.e., the consistency of $L(X, Y)$ with the commitment C, we could now rely on the consistency of the partial univariate interpolants. Originally, we had to derive the consistency of the bivariate interpolating polynomial $L(X, Y)$ directly from the consistency of the evaluations on some interpolation domain (established in the accepting proofs we got from the prover). Now, we could split the argument into the first step where we derive the consistency of the partial univariate interpolating polynomials $L_{x_i}(Y)$ and then derive the consistency of the bivariate $L(X, Y)$ from the consistency of the univariate polynomials $L_{x_i}(Y)$. Importantly, we could now proceed inductively via univariate interpolation in each step.

Let us have a set $U = \{x_i\}_{i \in [m]}$, where for each $x \in U$ we have that eq. (3) holds. First, we justify that, for all $x \in U$, we can assume $[c - L_x(\tau)]_1 \bullet [1]_2 = [\varphi]_1 \bullet [\sigma - x]_2$ when extracting from a polynomially bounded prover. Suppose that $[c - L_x(\tau)]_1 \bullet [1]_2 \neq [\varphi]_1 \bullet [\sigma - x]_2$. If we denote $\chi = c - L_x(\tau)$ and $b = \varphi(\sigma - x)$ then, by eq. (3), we managed to use the prover to efficiently construct elements $[\chi]_1 \neq [b]_1$, $[\varphi]_1$, and $[\psi]_1$ and a set S of size $n + 1$ such that $\psi = (\chi - b)/Z_S(\tau)$. Intuitively, this should be hard as our input is the commitment key ck where the maximal power of τ in any term is n, yet we managed to divide by the evaluation $Z_S(\tau)$ of a polynomial $Z_S(Y)$ of degree $n + 1$. The computational hardness of the corresponding task can be seen as a generalization of q-strong Diffie-Hellman (q-SDH) assumption similar to the ARSDH assumption of Lipmaa et al. [LPS24b]. Thus, under such computational hardness assumption, it holds that $[c - L_x(\tau)]_1 \bullet [1]_2 = [\varphi]_1 \bullet [\sigma - x]_2$ except with a negligible probability.

Now, we conclude by showing that $[c - L(\sigma, \tau)]_1 = [0]_1$ from the consistency of the univariate partial interpolants $L_{x_i}(Y)$. In the same way we showed eq. (3) from the $n+1$ identities described in eq. (2), we can use the $m+1$ identities $[c - L_{x_i}(\tau)]_1 \bullet [1]_2 = [\varphi_{x_i}]_1 \bullet [\sigma - x_i]_2$ to derive that

$$[c - L(\sigma, \tau)]_1 \bullet [1]_2 = [\varphi]_1 \bullet [Z_U(\sigma)]_2,$$

where $[\varphi]_1 = \sum_{i=0}^{m} d_i^U [\varphi_{x_i}]_1$. The derivation is a special case of the bivariate batching lemma we discussed above, and, in fact, it is exactly the univariate batching lemma from [LPS24a]. Additionally, similarly to above, finding group elements that satisfy this last equality is likely computationally hard for $[c - L(\sigma, \tau)]_1 \neq [0]_1$ as it would be equivalent to finding $[\chi]_1 = [c - L(\sigma, \tau)]_1 \neq [0]_1, [\varphi]_1$, and U of size $m+1$ such that $\varphi = \chi / Z_U(\sigma)$. In fact, the computational hardness of this task is exactly the ARSDH assumption of Lipmaa et al. [LPS24b]. Therefore, we can conclude that $[c - L(\sigma, \tau)]_1 = [0]_1$, showing that the interpolating polynomial $L(X, Y)$ is consistent with the commitment C.

Concluding about the direct interpolation via an interpolation shuffle. In summary, the above discussion establishes the special soundness of the bivariate KZG scheme under the *bivariate* ARSDH assumption (We show in Section 5.2 that bivariate ARSDH implies the original ARSDH.) via the above two stage interpolation from a set of accepting evaluation proofs $\{(C, (x_i, y_j), z_{i,j}, \pi_{i,j} = (\pi_i^{(1)}, \pi_{i,j}^{(2)}))\}_{x_i \in U, y_j \in S}$, i.e., where the evaluation proofs for any pair of evaluation points (x_i, y_j) and (x_i, y_k) share the first component $\pi_i^{(1)}$. Note that, by rearranging the terms, the corresponding interpolating polynomial $\sum_{i=0}^{m} \ell_i^U(X) L_{x_i}(Y)$ is equal to the polynomial we get by direct bivariate interpolation on the evaluation domain $U \times S$ for $U = \{x_i\}_{i \in [m]}$ and $S = \{y_j\}_{j \in [n]}$, i.e.,

$$L(X, Y) = \sum_{i=0}^{m} \ell_i^U(X) L_{x_i}(Y) = \sum_{i=0}^{m} \sum_{j=0}^{n} z_{i,j} \ell_i^U(X) \ell_j^S(Y).$$

Thus we have been implicitly discussing the direct interpolating extractor all along, and the above discussion, in fact, establishes the consistency of its output with the commitment C. In the following Section 2.2, we discuss how to extract transcripts with the needed structure.

Could there be a reduction from extractability of univariate KZG to extractability of (two-round) bivariate KZG? Note that the first element of the proof π_1 is a commitment to the first quotient polynomial $q_1(X, Y) = (f(X, Y) - f(x, Y))/(X - x)$, and, hence, the group element $C - (\sigma - x) \cdot \pi_0 = C - [f(\sigma, \tau) - f(x, \tau)]_1 = [f(x, \tau)]_1$ is a commitment to the *univariate* polynomial $f(x, Y)$. At first glance, one could hope to use only the univariate KZG extractor for interpolating $L(X, Y)$ as follows: first ask the prover for evaluation points $f(x, y_i)$ for $M+1$ distinct choices of y_i and extract the corresponding univariate polynomials $f(x_i, Y)$ and then simply interpolate the bivariate polynomial $f(X, Y)$ from these. Unfortunately, this extraction is not feasible, because it would require computing the commitment $C - (\sigma - x) \cdot \pi_o$, which, if successful, would allow solving the computational DH assumption on instance $(\pi_0, [\sigma - x]_1)$.

2.2 Extracting Structured Transcripts

To show that bivariate KZG is black-box extractable, we show how to extract a suitable set of structured accepting evaluation proofs from any prover who succeeds in providing an accepting evaluation proof with a non-negligible probability. Note that our arguments in Section 2.1 do not crucially rely on the product structure of the evaluation domain, and we used it only to aid the simplicity of exposition of our proof. The same consistency property holds under bivariate ARSDH also for the interpolating polynomial

$$L(X, Y) = \sum_{i=0}^{m} \sum_{j=0}^{n} z_{i,j} \ell_i^U(X) \ell_j^{S_i}(Y)$$

defined using an accepting set $\{(C, (x_i, y_{i,j}), z_{i,j}, \pi_{i,j} = (\pi_i^{(1)}, \pi_{i,j}^{(2)}))\}_{x_i \in U, y_{i,j} \in S_i}$, i.e., with a possibly distinct set $S_i = \{y_{i,0}, \ldots, y_{i,m}\}$ for each $x_i \in U$. Thus, it is sufficient to extract such a set of transcripts, which slightly simplifies our task.

The natural approach is to proceed via uniform rejection sampling as follows. At the i-th iteration, we sample a uniform $x_i \leftarrow \mathbb{F}$ and then sample $y_{i,j} \leftarrow \mathbb{F}$ until we collect $n+1$ accepting transcript from the prover w.r.t. x_i and distinct $y_{i,j}$'s. Then we sample new x_{i+1} and repeat. If the prover's success probability is δ then we expect to get the sought transcripts after roughly $(m+1)(n+1)\delta^{-2}$ samples with probability close to one.

However, the above extraction strategy does not ensure any consistency of the extracted accepting evaluation proofs. The main issue is that if we ask the prover for evaluation proofs for any pair of evaluation points (x_i, y_j) and (x_i, y_k) sharing the X-value then we are not guaranteed a pair of evaluation proofs $(C, (x_i, y_j), z_{i,j}, \pi_{i,j} = (\pi_{i,j}^{(1)}, \pi_{i,j}^{(2)}))$ and $(C, (x_i, y_k), z_{i,k}, \pi_{i,k} = (\pi_{i,k}^{(1)}, \pi_{i,k}^{(2)}))$ such that $\pi_{i,j}^{(1)} = \pi_{i,k}^{(1)}$. The question of whether we can extract such a pair of evaluation proofs is strongly related to the so-called *computational uniqueness of proofs* property of the bivariate KZG scheme, i.e., that finding multiple accepting proofs for the same statement should be computationally hard. On one hand, by our explicit polynomial decomposition lemma (Lemma 1), honest evaluation proofs for evaluation points sharing a prefix of the coordinates share also prefixes of the proof elements (each quotient polynomial depends only on the corresponding prefix of the evaluation point). Also, if computational uniqueness of proofs does not hold for the multivariate KZG scheme then we cannot hope to extract transcripts with consistent proofs from the prover. It was observed that *univariate* KZG scheme satisfies the computational uniqueness of proofs under the bilinear n-strong Diffie-Hellmann (n-SBDH) assumption, where n is the maximal degree of the committed univariate polynomial. However, the computational uniqueness of proofs does not hold for the bivariate KZG scheme in a very strong sense: given any accepting evaluation proof, it is possible to efficiently sample one of q distinct accepting proofs for the same statement, where q is the order of $\mathbb{G}_1$ (See Section 9 for an additional discussion of the computational uniqueness of proofs for the KZG family). Thus, a cheating prover can re-randomize evaluation proofs without any loss in its acceptance probability.

Proof consistency by extending the KZG PoKoP. To get around this issue and force the prover to provide consistent proofs for evaluation proofs sharing the X-coordinate, we consider an extended variant of the canonical PoKoP for the bivariate KZG scheme. Recall from the description of the bivariate KZG scheme that, given an evaluation point (x, y) the two components of an evaluation proof are simply commitments to evaluations of the quotient polynomials $q_1(X, Y) = \frac{f(X,Y)-f(x,Y)}{X-x}$ and $q_2(Y) = \frac{f(x,Y)-f(x,y)}{Y-y}$ at the secret evaluation point (σ, τ). Importantly, the polynomial q_1 does not depend on the Y-coordinate of the evaluation point and the prover can compute $\pi_1 = [q_1(\sigma, \tau)]_1$ once it receives x, i.e., the X-coordinate of the evaluation point. This observation allows us to perform the evaluation proof interactively: The verifier first sends *only* the X-coordinate of the evaluation point to the prover. The prover computes the first proof element π_1 and sends it to the verifier. Then, the verifier sends also the Y-coordinate to the prover who can now compute $z = f(x, y)$ and π_2, and send them to the verifier. Finally, the verifier performs the same check as in the bivariate KZG scheme. We call this protocol the *extended* PoKoP for the bivariate KZG scheme.

Even though seemingly insignificant, the move to the extended KZG PoKoP allows us to use rewinding to extract evaluation proofs that share the first component for evaluation points sharing the X-coordinate. If the extractor proceeds with the two-round evaluation proof with $(x_i, y_{i,j})$ it receives $\pi_{i,j}^{(1)}$ in the first round and $\pi_{i,j}^{(2)}$ in the second. Then, it can rewind the prover to the state after the first round and run the second round with Y-coordinate $y_{i,k}$, receiving $\pi_{i,k}^{(2)}$. In case both $(\pi_{i,j}^{(1)}, \pi_{i,j}^{(2)})$ and $(\pi_{i,j}^{(1)}, \pi_{i,k}^{(2)})$ are accepting, we indeed have accepting evaluation proofs that share the first component for evaluation points $(x_i, y_{i,j})$ and $(x_i, y_{i,k})$. We use the rewinding approach to show that if the success probability of the prover over $(x, y) \leftarrow \mathbb{F}^2$ is δ then, with constant probability we can extract the $T = (m+1)(n+1)$ accepting transcripts needed for special soundness after performing roughly $T\delta^{-2}$ evaluation proofs. For an inverse polynomial δ, this would result in a polynomial-time extraction.

At first, it might be somewhat confusing that the prover learns the complete evaluation point only in the last round of the extended PoKoP, and, in fact, we crucially rely on this property towards extraction as we discuss above. However, note that our original goal was to establish the extractability of a PoKoP, i.e., a proof of *knowledge of a polynomial*, and, thus, there is no specific evaluation point we really wish to evaluate the commited polynomial on. Nevertheless, through the extended protocol, the verifier would still learn an evaluation of the committed polynomial at a uniform evaluation point and could use it outside of the scope of the extended PoKoP when used as a subroutine in some higher-level protocol (e.g., in a PIOP to SNARK compiler). Finally, note that the extended protocol is, for all practical purposes, equivalent to the canonical one, where the verifier sends the complete evaluation point in a single round. We did not introduce any overhead to the parties and the protocol is still public-coin. Thus, in the Random Oracle Model, we could use the Fiat-Shamir heuristic in a round-by-round fashion to collapse it into a non-interactive PoKoP.

By combining the observations from Sections 2.1 and 2.2 we establish black-box extractability of the bivariate KZG scheme:

Theorem 1. *(informal) Under the bivariate* ARSDH *assumption, the extended* PoKoP *is a proof of knowledge of a polynomial for the bivariate KZG scheme.*

In the following sections, we discuss how we extend this result to the multivariate scheme (Section 2.3) and to the *canonical* PoKoP (Section 2.4).

2.3 Extractability of Multivariate KZG

Our proof of black-box extractability for the bivariate KZG can be naturally extended to the n-variate KZG scheme from [PST13] and its other recent variants such as [LXZ+24, LLZ+24].

First, we recall the multivariate scheme using our explicit polynomial decomposition lemma (Lemma 1), which shows that for any multivariate polynomial $f(X_1, \ldots, X_n)$, the prover can prove the correctness of evaluation at $(x_1, \ldots, x_n)$ using the following identity

$$f(X_1, \ldots, X_n) - f(x_1, \ldots, x_n) = \sum_{i=1}^{n} q_i(X_i, \ldots, X_n)(X_i - x_i),$$

where $q_i(X_i, \ldots, X_n) = \frac{f(x_1,\ldots,x_{i-1},X_i,\ldots,X_n) - f(x_1,\ldots,x_i,X_{i+1},\ldots,X_n)}{(X_i - x_i)}$ is a polynomial in variables $X_i, \ldots, X_n$ for all $1 \leq i \leq n$. The commitment to f is $C = [f(t_1, \ldots, t_n)]_1$ for a secret evaluation point $(t_1, \ldots, t_n)$. Analogously to the bivariate case, the commitment key ck contains the evaluations of all the n-variate monomials satisfying some degree bounds $\mathbf{d} = (d_1, \ldots, d_n)$ on $(t_1, \ldots, t_n)$. The evaluation proof π comprises commitments to the n quotient polynomials, i.e., $\pi = (\pi_1, \ldots, \pi_n) = ([q_1(t_1, \ldots, t_n))]_1, \ldots, [q_n(t_n)]_1)$. For an evaluation point $(x_1, \ldots, x_n)$ and the claimed value z, the verifier checks that

$$(C - [z]_1) \bullet [1]_2 = \bigotimes_{i=1}^{n} \pi_i \bullet [t_i - x_i]_2$$

given the commitment C, evaluation proof π and commitment key ck.

Special soundness. Similarly to the proof of special soundness for the bivariate KZG scheme under the bivariate ARSDH assumption outlined in Section 2.1, the core technical result is an n-variate batching lemma (see Lemma 2 in Section 5.1). Let $d_1, \ldots, d_n$ be the maximal degrees for the variables $X_1, \ldots, X_n$ in f (respectively). Then, given $d_n + 1$ accepting proofs for the evaluation points $\{(x_1, \ldots, x_{n-1}, y_j)\}_{y_j \in S}$ with $S = \{y_j\}_{j \in [d_n]}$, we have $d_n + 1$ equations

$$[c - z_j]_1 \bullet [1]_2 = \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2 \otimes [\psi_j]_1 \bullet [t_n - y_j]_2. \tag{4}$$

Again, we can establish that

$$[c - L_{x_1,\ldots,x_{n-1}}(t_n)]_1 \bullet [1]_2 = \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2 \otimes [\psi]_1 \bullet [Z_S(t_n)]_2 \tag{5}$$

for $L_{x_1,\ldots,x_{n-1}}(X_n) = \sum_{j=0}^{d_n} z_j \ell_j^S(X_n)$ and $[\psi]_1 = \sum_{j=0}^{d_n} d_j^S [\psi_j]_1$. If it additionally holds that

$$[c - L_{x_1,\ldots,x_{n-1}}(t_n)]_1 \bullet [1]_2 = \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2 \tag{6}$$

then we have an evaluation proof in an $(n-1)$-variate KZG scheme certifying that the $(n-1)$-variate polynomial $f_{t_n}(X_1, \ldots, X_{n-1}) = f(X_1, \ldots, X_{n-1}, t_n)$ evaluates to $L_{x_1,\ldots,x_{n-1}}(t_n)$ at $(x_1, \ldots, x_{n-1})$. Note that Equation (6) would be implied by the corresponding n-variate ARSDH, which we denote ARSDH(n). Thus, we can proceed inductively over the variables, at the i-th step relying on ARSDH(i). Specifically, we show that if we interpolate over a feasible interpolation domain using accepting transcripts that share prefixes of the evaluation proofs then the interpolating polynomial is consistent with the commitment C except with negligible probability under the n-variate ARSDH.

From special soundness to knowledge-soundness. Note that, for an arbitrary number of variables n, each quotient polynomial q_i depends only on $(x_1, \ldots, x_i)$. Analogously to the bivariate case, the verifier can submit only the i-th coordinate of the evaluation point $(x_1, \ldots, x_n)$ in the i-th round, which is sufficient for the prover to compute the i-th element of the evaluation proof. This gives rise to the *extended* PoKoP for the multivariate KZG scheme, for which we can rewind the prover when extracting transcripts of evaluation proofs that share prefixes of coordinates in the evaluation points and the corresponding elements of the evaluation proofs. However, for super-constant number of variables, and, hence, number of rounds in the extended PoKoP, we must face the common bottleneck of rewinding for multi-round protocols. The naive analysis of success probability would result in an extraction probability of roughly δ^n, which is negligible for a super-constant number of variables n. To avoid such steep degradation, we rely on a stronger forking lemma established by Attema, Cramer, and Kohl [ACK21]. Thus, we prove black-box extractability of the bivariate KZG scheme:

Theorem 2. *(informal) Under the n-variate* ARSDH *assumption, the extended* PoKoP *is a proof of knowledge of a polynomial for the n-variate KZG scheme.*

Towards the proof, we truly reap the benefits of the framework of PoKoPs. Since we established the (computational) special soundness of the extended PoKoP as a standard interactive proof for the commitment relation of the multivariate KZG scheme, we can invoke the [ACK21] extractor as a black box in our proof of knowledge-soundness. Notably, in their proof of extractability for the univariate KZG scheme w.r.t. the definition of black-box extractability tailored to polynomial commitment schemes, [LPS24a] had to adapt the ACK extractor and reprove the guarantees on its performance. We discuss the relationship of PoKoPs to the notion of black-box extractability for polynomial commitment schemes from [LPS24a] in detail in Section 6.3. Importantly, it is possible to show that our results imply extractability of the multivariate KZG scheme under their definition.

2.4 General Interpolating Extractors without Rewinding

In Section 5.4, we show a generalized batching lemma (Lemma 5) that allows for a more direct proof of the consistency of the interpolated polynomial $L(X_1, \ldots, X_n)$ with the commitment C under the corresponding generalization of ARSDH(n), which we call GARSDH(n). There are two crucial differences between the *generalized* multivariate batching lemma and the multivariate batching lemma we discussed in Section 2.3. Firstly, we batch $(d_1+1)\cdots(d_n+1)$ evaluation proofs for the whole evaluation domain simultaneously, which allows

us to directly derive an identity for $[c - L(t_1, \ldots, t_n)]_1$. Secondly, there is no need for a consistency requirement among the evaluation proofs for pairs of evaluation points that share a prefix of coordinates. Therefore, we do not need to rewind the prover during the evaluation proof and we can prove the knowledge-soundness of the canonical PoKoP for the multivariate KZG scheme.

Theorem 3. *(informal) Under the n-variate* GARSDH *assumption, the canonical* PoKoP *is a proof of knowledge of a polynomial for the n-variate KZG scheme.*

On the other hand, the identity derived via batching all the evaluation proofs for the whole interpolation domain at once is much more structured. Thus, the corresponding hardness assumption GARSDH(n) one needs to postulate is arguably more tailored to facilitating extraction via interpolation compared to ARSDH(n). In some sense, the proof of extractability of the canonical PoKoP under GARSDH illustrates that proof-consistency is not essential for proving the extractability of the multivariate KZG scheme rather than providing a better result compared the extractability of the extended KZG PoKoP. Indeed, the ARSDH assumption is plausibly weaker than GARSDH.

2.5 Applying our Framework to Other KZG Variants

Ultimately, our results exploit only the basic structure of the verification check which is common in most variants of the KZG scheme, and, thus, they can be applied rather directly as long as the particular scheme is based on a KZG-like verification check as we demonstrate in Section 7.

First, we show that as a corollary of our results for the multivariate KZG scheme, we get extractability for the bivariate KZG variant from [LXZ+24] and the multivariate KZG variant from [LLZ+24]. Both schemes change the proof generation algorithm to support distributed proof generation. Importantly, the verifier can be completely oblivious to the particular way of computing the proofs. The proof π is still a vector of group elements and the verification check is the same as in multivariate KZG, and, thus, we can directly apply our results.

Additionally, we revisit the *randomized* multivariate KZG scheme [PST13], which was introduced to deal with some potential deficiencies of the basic multivariate scheme with respect to evaluation binding (see Section 10 for an extensive discussion). Importantly, even though the proof structure is the same, the verification check in the randomized scheme is significantly different as presented in [PST13]. However, we show that the scheme can be presented equivalently such that both the proof structure and the verification check are the same as in the basic multivariate KZG scheme. Therefore, we get the extractability for the randomized multivariate KZG scheme as well.

2.6 Analysis of ARSDH and GARSDH in Idealized Models

In Section 8, we analyze ARSDH and GARSDH in the Algebraic Group Model (AGM) [FKL18] and the Generic Group Model (GGM) [Sho97]. First, we show that in AGM both assumptions imply the Power Discrete Logarithm (PDL) assumption. Besides providing the minimal sanity check when postulating the new assumptions, the AGM reductions from ARSDH and GARSDH to PDL to some extent validate the specific phrasing of the winning conditions in ARSDH and GARSDH. On the other hand, the reductions to PDL in the AGM do not provide a way of distinguishing between ARSDH and GARSDH in terms of their relative hardness. Additionally, we prove query complexity lower bounds for ARSDH and GARSDH in the GGM supporting the intuition that ARSDH is plausibly weaker than GARSDH.

3 Preliminaries

For $n, m \in \mathbb{N}, n < m$ we denote by $[n, m]$ the set $[n, m] = \{n, n+1, \ldots, m\}$ and we denote $[n] = [0, n]$. For a finite set S we denote by $x \leftarrow S$ that x is a uniformly distributed sample from S. We say that a function $\varepsilon \colon \mathbb{N} \to \mathbb{R}^+$ is negligible if it approaches zero faster than any inverse polynomial. In that case, we write $\varepsilon(\lambda) \in \mathsf{negl}(\lambda)$.

Definition 1 (Vanishing Polynomial, Lagrange Basis, and Lagrange weights). *For any set* $S = \{a_i\}_{i\in[t]} \subseteq \mathbb{F}$ *such that* $a_i \neq a_j$ *for all* $i \neq j$*, we define* the Vanishing polynomial $Z_S(X)$ of S *as* $Z_S(X) = \prod_{i\in[t]}(X - a_i)$ *and the* i-th Lagrange polynomial $\ell_i^S(X)$ of S*, respectively the* i-th Lagrange weight w_i^S of S*, as*

$$\ell_i^S(X) = \prod_{j\in[t],j\neq i} \frac{X - a_i}{a_i - a_j}, \quad \textit{respectively} \quad w_i^S = \frac{1}{Z_{S\setminus\{a_i\}}(a_i)} = \frac{1}{\prod_{j\in[t],j\neq i}(a_i - a_j)}.$$

Notice that, using the notation from Definition 1, it holds for all $i \in [t]$ that $\ell_i^S(X)(X - a_i) = Z_S(X)w_i^S$.

3.1 Bilinear Groups and Pairings

Consider two additive groups $(\mathbb{G}_1, +_1)$ and $(\mathbb{G}_2, +_2)$ and a multiplicative group $(\mathbb{G}_T, \otimes)$, all of prime order p. A *pairing* $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ satisfies:

Bilinearity: For all $P_1 \in \mathbb{G}_1, P_2 \in \mathbb{G}_2$, and $a, b \in \mathbb{F}_p$, it holds that $e(aP_1, bP_2) = e(P_1, P_2)^{ab}$.
Non-degeneracy: For all $P_1 \neq 0_{\mathbb{G}_1}$ and $P_2 \neq 0_{\mathbb{G}_2}$, it holds that $e(P_1, P_2) \neq 1_{\mathbb{G}_T}$,

where $0_{\mathbb{G}_1}$ (resp. $0_{\mathbb{G}_2}$ and $1_{\mathbb{G}_T}$) is the neutral element of $\mathbb{G}_1$ (resp. $\mathbb{G}_2$ and $\mathbb{G}_T$).

To simplify notation and make algebraic manipulations clearer, we use the bracket notation for group elements, where we write $[a]_1 = a[1]_1$, where $[1]_1$ is the generator of $\mathbb{G}_1$ (and similarly for the other two groups). We use the symbol $\bullet$ as a binary operator that represents an application of the pairing, i.e., $[a]_1 \bullet [b]_2 = e([a]_1, [b]_2)$. The bilinearity property then, for example, gives us

$$[a]_1 \bullet [b]_2 = ([1]_1 \bullet [1]_2)^{ab} = [ab]_1 \bullet [1]_2 = ([a]_1 \bullet [1]_2)^b.$$

We assume that there is some *bilinear group generator* PGen that samples the groups $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$, and outputs the corresponding *group parameters* $\mathsf{gp} = (p, [1]_1, +_1, [1]_2, +_2, [1]_T, \otimes, \bullet)$ that specify the order of the groups, and allow to compute the respective group operations and pairing efficiently. In the rest of the paper, we simply write $\mathsf{gp} = (p, [1]_1, [1]_2, \bullet)$ for brevity.

3.2 Relations and Interactive Proofs

A relation $\mathcal{R}$ is a set of pairs (x, w). We call x the *statement* and w the *witness*. An *interactive proof* allows a prover P to prove to a verifier V that there exists a witness w for statement x such that $(x, w) \in \mathcal{R}$.

Definition 2 (Interactive Proof). *For a function* $\varepsilon : \mathbb{N} \to [0, 1]$*, an* interactive proof for relation $\mathcal{R}$ *is a pair of interactive* PPT *algorithms* (P, V) *satisfying the following properties:*

Completeness: *For every* $(x, w) \in \mathcal{R}$*, if* V *interacts with* P *on the common input* x *and* P *has private input* w*, then* V *accepts with probability* 1.
Soundness: *For every* x *that does not have a witness* w *in* $\mathcal{R}$ *and every (computationally unbounded) cheating prover strategy* $\widetilde{\mathsf{P}}$*, the verifier* V *accepts when interacting with* $\widetilde{\mathsf{P}}$ *with probability less than* $\varepsilon(|x|)$*, where* ε *is called the* soundness error.

When restricting the set of cheating provers to PPT *algorithms, we use the term* interactive argument *instead.*

A *proof of knowledge* is an interactive proof that demonstrates not only the *existence* of a witness w for x but that the P also *knows* this witness.

Definition 3 (Proof/Argument of Knowledge and (Computational) Knowledge-Soundness). *A* proof of knowledge *is an interactive proof* (P, V) *that is* knowledge sound*, i.e., if there exists an expected polynomial time extractor* Ext *with black-box rewinding access to the prover such that, for all instances* x *and any (computationally unbounded) prover strategy* P^* *that makes* V *accept on* x *with probability* δ*, it holds that*

$$\Pr\left[(x, w) \in \mathcal{R} \mid w \leftarrow \mathsf{Ext}^{\mathsf{P}^*}(x)\right] \geq \frac{\delta - \varepsilon}{\mathsf{poly}(|x|)},$$

where poly *is some positive polynomial and* $\varepsilon \in [0,1]$ *is the* knowledge error.

In an argument of knowledge *we restrict the set of cheating provers to* PPT *algorithms. Additionally, we treat both the success probability of* P^* *and soundness error as functions of* $|x|$ *and we require negligible knowledge error.*

We note that in [WTS+18], it was shown that generalized special soundness implies witness-extended emulation [Lin03], a property that is often essential when using a proof of knowledge as a subprotocol within a higher-level protocol.

One way of proving knowledge-soundness is to first show that the proof is *special sound*, i.e., that given a sufficient number of accepting transcripts for statement x arranged in a tree structure, one can extract the corresponding witness. If this *tree of transcripts* can be constructed efficiently given access to $\widetilde{\mathsf{P}}$, the proof is knowledge sound.

Definition 4 (Tree of Transcripts). *A* $(k_1, \ldots, k_n)$-tree of transcripts for an n-round interactive protocol *is a set of* $\prod_{i=1}^{n} k_i$ *transcripts of the form* $(x, c_1, \ldots, c_n, a_1, \ldots, a_n)$*, where* x *denotes the instance,* $c_1, \ldots, c_n$ *the verifier's messages, and* $a_1, \ldots, a_n$ *the prover's messages. The transcripts admit the following tree structure. Each transcript corresponds to exactly one path from the root node to a leaf node. The tree's root (at level 1) is labelled by the instance* x*, the remaining vertices on the path to a leaf are labelled by the* a_i*'s, where vertex on level* $i+1$ *gets the label* a_i*. The edges are labelled by the* c_i*'s, where the edge incident to vertices on levels* i *and* $i+1$ *gets the label* c_i*. Each vertex at the* i*-th level of the tree has precisely* k_i *children, corresponding to* k_i *distinct choices for* c_i *that label the edges.*

The standard definition of generalized $(k_1, \ldots, k_n)$-special soundness requires extraction of a witness from an arbitrary $(k_1, \ldots, k_n)$-tree of accepting transcripts. In this work, we consider its *computational* variant, where we require the extraction of the witness from an arbitrary $(k_1, \ldots, k_n)$-tree of accepting transcripts produced via some PPT algorithm.

Definition 5 (Computational $(k_1, \ldots, k_n)$-Special Soundness). *An interactive proof* (P, V) *for relation* $\mathcal{R}$ *is* computationally $(k_1, \ldots, k_n)$-special sound *if there exists a* PPT *extractor* Ext *such that for all* PPT *algorithms* Tree *it holds that*

$$\Pr\left[(x,w) \in \mathcal{R} \;\middle|\; \begin{array}{l} \mathcal{T} \leftarrow \mathsf{Tree}(x, k_1, \ldots, k_n), \\ \mathcal{T} \text{ is a } (k_1, \ldots, k_n)\text{-tree of accepting transcripts for } (\mathsf{P}, \mathsf{V}), \\ w \leftarrow \mathsf{Ext}(x, \mathcal{T}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

3.3 Polynomial Commitment Schemes

A Polynomial Commitment Scheme (PCS) is a cryptographic primitive that allows a prover to commit to a polynomial and later reveal its evaluations at specific points, along with a proof of correctness. Next, we define the class of *Non-Interactive Multivariate Polynomial Commitment Scheme*, consisting of the following algorithms: KGen, Com, Open, and Ver. These algorithms enable efficient commitment, evaluation, and verification of multivariate polynomials with specified degree bounds.

Definition 6 (Multivariate Polynomial Commitment Scheme). *A* Non-Interactive Multivariate Polynomial Commitment Scheme *is a tuple of algorithms* $(\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ *such that:*

$\mathsf{KGen}(1^\lambda, \mathbf{d}, \mathsf{aux})$**:** *Given a security parameter* 1^λ*, a vector of degree bounds* $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{N}^n$*, and an auxiliary input* aux*, returns a* commitment key ck.

$\mathsf{Com}(\mathsf{ck}, f)$**:** *Given a commitment key* ck *and a polynomial* $f \in \mathbb{F}[X_1, \ldots, X_n]^{\leq \mathbf{d}}$*, returns a* commitment C.

$\mathsf{Open}(\mathsf{ck}, C, f, \mathbf{x})$**:** *Given a commitment key* ck*, a commitment* C*, a polynomial* $f \in \mathbb{F}[X_1, \ldots, X_n]^{\leq \mathbf{d}}$*, and an* evaluation point $\mathbf{x} = (x_1, \ldots x_k) \in \mathbb{F}^n$*, returns* (z, π)*, where* $z = f(\mathbf{x})$ *and* π *is an* evaluation proof.

$\mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi)$**:** *Given a commitment key* ck*, a commitment* C*, an evaluation point* $\mathbf{x} = (x_1, \ldots x_n) \in \mathbb{F}^n$*, a proposed evaluation* $z \in \mathbb{F}$*, and an evaluation proof* π*, returns* 1 *(accept) or* 0 *(reject).*

The above Definition 6 is general. However, our focus is on pairing-based polynomial commitments. Note that the auxiliary input aux to KGen can accommodate the passing of the group parameters gp sampled by a bilinear group generator PGen or another algorithm PGen specifying the context the specific polynomial commitment is based on. To simplify notation, we assume that ck implicitly contains the degree bounds $\mathbf{d} = (d_1, \ldots, d_n)$ that were used to compute it, and the algorithms Com and Open can efficiently check that the input polynomial satisfies the appropriate degree bounds. We denote by $\mathbf{d}_{\mathsf{ck}}$ the degree bounds supported by the commitment key ck.

4 PoKoPs and the KZG PCS Family

In Section 4.1, we introduce Proofs of Knowledge of a Polynomial (PoKoPs), which, in the rest of the paper, we use to study the extractability of the KZG PCS family through the lens of classical proofs of knowledge. In Section 4.2, we state and prove an explicit variant of the multivariate polynomial decomposition lemma by Papamanthou, Shi and Tamassia [PST13] that is the basis for their generalization of the original univariate scheme by Kate, Zaverucha and Goldberg [KZG10] to multivariate polynomials. In Section 4.3, we present the KZG family of polynomial commitment schemes capturing both the univariate and multivariate KZG schemes. Finally, in Section 4.4, we present the two KZG PoKoPs we analyze in the subsequent sections.

4.1 Proof of Knowledge of a Polynomial

The basic security property of a polynomial commitment scheme is *evaluation binding*, which, roughly speaking, ensures that the prover is *committed to a function*, i.e., that it is computationally hard to produce a commitment C and accepting proofs π and π' for distinct evaluations z and z' w.r.t. the same evaluation point $\mathbf{x}$ (For the formal definition, see Section 10).

Evaluation binding might be sufficient in some applications. However, in the context of zk-SNARKs, it is crucial that the polynomial commitment scheme ensures the prover is, in fact, *committed to a polynomial* of the correct degree, a requirement captured by *knowledge-soundness*. Intuitively, a PCS is knowledge sound if any prover that makes the verification algorithm Ver accept an evaluation z knows a polynomial that is consistent with the commitment C and the evaluation. There are various definitions of the knowledge-soundness property for polynomial commitment schemes in the literature. Below, we define a *Proof of Knowledge of a Polynomial* (PoKoP), which guarantees a notion of extractability for polynomial commitment schemes sufficient for their common application in constructions of zk-SNARKs.

Recall that, for a commitment key ck, we denote by $\mathbf{d}_{\mathsf{ck}}$ the vector of degree bounds supported by ck. Additionally, for a polynomial $f \in \mathbb{F}_q[X_1, \ldots, X_n]$, we denote by $\overline{\deg}(f)$ the vector of maximal degrees of the variables in F, i.e., $\overline{\deg}(f) = (\deg_{X_1}(f), \ldots, \deg_{X_n}(f))$. The following relations capture when a polynomial f is consistent with both the commitment C and the degree bounds $\mathbf{d}_{\mathsf{ck}}$. Additionally, the second relation captures also the consistency of f with a value z at an evaluation point $\mathbf{x}$.

Definition 7 (Commitment Relations). *Let* $\mathsf{PCS} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ *be a polynomial commitment scheme and* ck *a commitment key. We define the corresponding* commitment relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS}}$ *as*

$$\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS}} = \left\{(C, f) \mid C = \mathsf{PCS.Com}(\mathsf{ck}, f) \wedge \overline{\deg}(f) \leq \mathbf{d}_{\mathsf{ck}}\right\},$$

and the commitment evaluation relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS},eval}$ *as*

$$\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS},eval} = \left\{((C, \mathbf{x}, z), f) \mid C = \mathsf{PCS.Com}(\mathsf{ck}, f) \wedge \overline{\deg}(f) \leq \mathbf{d}_{\mathsf{ck}} \wedge f(\mathbf{x}) = z\right\}.$$

Now, we can define PoKoP simply as an argument of knowledge for the commitment relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS}}$.

Definition 8 (Proof of Knowledge of a Polynomial). *An interactive protocol* (P, V) *is a* proof of knowledge of a polynomial *for a polynomial commitment scheme* PCS *w.r.t. the parameter generator* PGen *and degree bounds* $d_1, \ldots, d_n \in \mathbb{N}$ *if it is an argument of knowledge for the commitment relation* $\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS}}$ *when* $\mathsf{ck} \leftarrow \mathsf{PCS.KGen}(1^\lambda, \mathbf{d} = (d_1, \ldots, d_n), \mathsf{aux})$ *for* $\mathsf{aux} \leftarrow \mathsf{PGen}(1^\lambda)$.

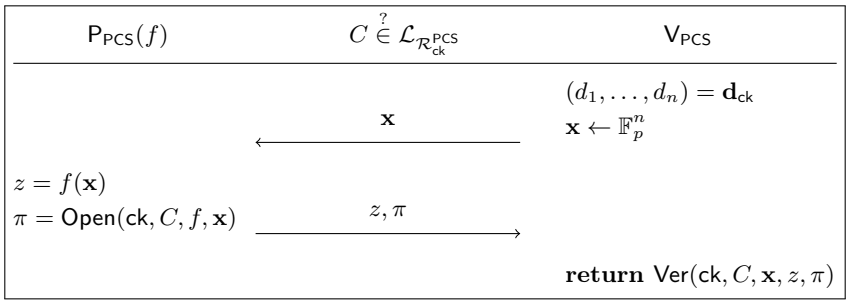

Fig. 1. The canonical PoKoP $(\mathsf{P}_{\mathsf{PCS}}, \mathsf{V}_{\mathsf{PCS}})$ of a $\mathsf{PCS} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$.

Instead of an argument of knowledge for the commitment relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS}}$, some previous works define knowledge-soundness of a polynomial commitment scheme basically as our notion of PoKoP but w.r.t. the commitment *evaluation* relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS},\mathrm{eval}}$ (see, e.g., [Lib24, LLZ+24, LXZ+24, PP24, Lee21, BFS19] and the references therein). Specifically, they would require that the (non-)interactive protocol $(\mathsf{PCS.Open}, \mathsf{PCS.Ver})$ is an argument of knowledge for $\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS},\mathrm{eval}}$ in the vein of Definition 8. Such choice, however, does not seem to be motivated by the need for extraction from an evaluation proof for an arbitrary evaluation point $\mathbf{x}$, but rather by the need to use the evaluation z in some higher-level compiler that uses the PCS as a building block. In fact, the standard use of polynomial commitments in many higher-level protocols is via a canonical challenge-response protocol defined next, where the prover sends an evaluation together with an evaluation proof for a *uniformly random evaluation point* $\mathbf{x}$ sampled by the verifier.

Definition 9 (Canonical PoKoP of PCS). *Let* $\mathsf{PCS} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ *be a polynomial commitment scheme. We call the protocol* $(\mathsf{P}_{\mathsf{PCS}}, \mathsf{V}_{\mathsf{PCS}})$ *defined in Figure 1 the* canonical PoKoP of PCS.

Note that the canonical PoKoP is w.r.t. to the commitment relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{PCS}}$, but, due to its specific structure, the verifier learns an evaluation z of f on the uniform evaluation challenge $\mathbf{x}$. Thus, the verifier can still use the evaluation z in any higher-level protocol.

4.2 Explicit Quotient Decomposition for Multivariate Polynomials

At the core of the extremely efficient KZG evaluation proof is the following basic proposition about multivariate polynomials: a multivariate polynomial $g(X_1, \ldots, X_n)$ is contained in the ideal generated by the linear univariate polynomials $(X_1 - x_1), \ldots, (X_n - x_n)$ if and only if $g(x_1, \ldots, x_n) = 0$. Thus, given $f(X_1, \ldots, X_n)$, $(x_1, \ldots, x_n)$, and z, we can test whether $f(x_1, \ldots, x_n) = z$ by verifying that the polynomial $g(X_1, \ldots, X_n) = f(X_1, \ldots, X_n) - f(x_1, \ldots, x_n)$ can be generated as a polynomial combination of $(X_1 - x_1), \ldots, (X_n - x_n)$. Such a test would be nonsensical when the polynomial is given explicitly, as we could simply evaluate f at $(x_1, \ldots, x_n)$ and compare the result to z. However, it turns out to be useful when working with an implicit representation of f such as a commitment.

In the following lemma, we give an explicit decomposition of $f(X_1, \ldots, X_n) - f(x_1, \ldots, x_n)$ as a sum of quotient polynomials, which is useful for presenting the KZG PCS family as well as in our proofs. The proof of the lemma follows by a careful analysis of sequential polynomial division of f by the linear polynomials $(X_i - x_i)$.

Lemma 1. *Let* $f(X_1, \ldots, X_n) \in \mathbb{F}_p[X_1, \ldots, X_n]$ *be an* n*-variate polynomial. For all* $(x_1, \ldots, x_n) \in \mathbb{F}_p^n$*, it holds that*

$$f(X_1, \ldots, X_n) = \sum_{i=1}^{n} q_i(X_i, \ldots, X_n)(X_i - x_i) + f(x_1, \ldots, x_n),$$

where

$$q_i(X_i, \ldots, X_n) = \frac{f(x_1, \ldots, x_{i-1}, X_i, \ldots, X_n) - f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n)}{(X_i - x_i)}$$

is a polynomial in $\mathbb{F}_p[X_i, \ldots, X_n]$ *for all* $1 \leq i \leq n$.

Proof. First, we prove the existence of such q_i's. We start by dividing the polynomial $f(X_1, \ldots, X_n)$ by $(X_1 - x_1)$. This gives us

$$f(X_1, \ldots, X_n) = q_1(X_1, \ldots, X_n)(X_1 - x_1) + r_1(X_2, \ldots, X_n), \tag{7}$$

where r_1 is the remainder after polynomial division. Since we are dividing by the univariate polynomial $(X_1 - x_1)$, we can view the division as being done in the univariate polynomial ring $(\mathbb{F}[X_2, \ldots, X_n])[X_1]$. In particular, the remainder r_1 is a polynomial in X_1 with coefficients from $\mathbb{F}[X_2, \ldots, X_n]$ satisfying $\deg_{X_1}(r_1) < \deg_{X_1}(X_1 - x_1) = 1$. Thus, there is no variable X_1 in r_1.

If we continue by dividing the remainder then, in step i, we have

$$r_{i-1}(X_i, \ldots, X_n) = q_i(X_i, \ldots, X_n)(X_i - x_i) + r_i(X_{i+1}, \ldots, X_n), \tag{8}$$

where q_i only contains variables $X_i, \ldots, X_n$, and, similarly, r_i contains only the variables $X_{i+1}, \ldots, X_n$. After n polynomial divisions, we get the decomposition

$$f(X_1, \ldots, X_n) = \sum_{i=1}^{n} q_i(X_i, \ldots, X_n)(X_i - x_i) + r_n.$$

Next, we show that $r_i(X_{i+1}, \ldots, X_n) = f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n)$ for all $i \in [1, n]$ by induction. To show the base case for $i = 1$, we subtract $f(x_1, X_2, \ldots, X_n)$ from both sides of eq. (7), getting

$$f(X_1, \ldots, X_n) - f(x_1, X_2, \ldots, X_n) = q_1(X_1, \ldots, X_n)(X_1 - x_1) + r_1(X_2, \ldots, X_n) - f(x_1, X_2, \ldots, X_n).$$

If we evaluate the above equality at $X_1 = x_1$, we get $0 = r_1(X_2, \ldots, X_n) - f(x_1, X_2, \ldots, X_n)$, proving the base case.

To show the inductive step, we use the inductive hypothesis $r_{i-1}(X_i, \ldots, X_n) = f(x_1, \ldots, x_{i-1}, X_i, \ldots, X_n)$. From both sides of eq. (8), we subtract $f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n)$, getting

$$\begin{aligned} r_{i-1}(X_i, \ldots, X_n) - f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n) = \\ q_i(X_i, \ldots, X_n)(X_i - x_i) + r_i(X_{i+1}, \ldots, X_n) - f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n), \end{aligned}$$

which, by the inductive hypothesis, is equivalent to

$$\begin{aligned} f(x_1, \ldots, x_{i-1}, X_i, \ldots, X_n) - f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n) = \\ q_i(X_i, \ldots, X_n)(X_i - x_i) + r_i(X_{i+1}, \ldots, X_n) - f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n). \end{aligned}$$

Evaluating the above at $X_i = x_i$ gives $0 = r_i(X_{i+1}, \ldots, X_n) - f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n)$, proving the inductive step.

Importantly, $r_n = f(x_1, \ldots, x_n)$. Thus, we get

$$f(X_1, \ldots, X_n) = \sum_{i=1}^{n} q_i(X_i, \ldots, X_n)(X_i - x_i) + f(x_1, \ldots, x_n).$$

Finally, by rearranging eq. (8), we get

$$\begin{aligned} q_i(X_i, \ldots, X_n) &= \frac{r_{i-1}(X_i, \ldots, X_n) - r_i(X_{i+1}, \ldots, X_n)}{(X_i - x_i)} \\ &= \frac{f(x_1, \ldots, x_{i-1}, X_i, \ldots, X_n) - f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n)}{(X_i - x_i)}, \end{aligned}$$

which proves the claimed structure of the polynomials q_i. □

$\mathsf{KZG.KGen}(1^\lambda, \mathbf{d} = (d_1, \ldots, d_n), \mathsf{aux} = \mathsf{gp} = (p, [1]_1, [1]_2, \bullet))$:
Sample $t_1, \ldots, t_n \leftarrow \mathbb{F}_p$.
Output $\mathsf{ck} = \left(\mathsf{gp}, \left([t_1^{i_1} \cdots t_n^{i_n}]_1\right)_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, \left([t_i]_2\right)_{i=1}^{n}\right)$.
$\mathsf{KZG.Com}(\mathsf{ck}, f)$:
Output $C = [f(t_1, \ldots, t_n)]_1$.
$\mathsf{KZG.Open}(\mathsf{ck}, f, (x_1, \ldots, x_n))$:
Compute $z = f(x_1, \ldots, x_n)$.
For all $1 \leq i \leq n$, compute

$$q_i(X_i, \ldots, X_n) = \frac{f(x_1, \ldots, x_{i-1}, X_i, \ldots, X_n) - f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n)}{(X_i - x_i)}$$

and $\pi_i = [q_i(t_i, \ldots, t_n)]_1$.
Output $(\pi = (\pi_1, \ldots, \pi_n), z)$.
$\mathsf{KZG.Ver}(\mathsf{ck}, C, (x_1, \ldots, x_n), z, \pi = (\pi_1, \ldots, \pi_n))$:
If $(C - [z]_1) \bullet [1]_2 = \bigotimes_{i=1}^{n} (\pi_i \bullet [t_i - x_i]_2)$ then output 1, and 0 otherwise.

Fig. 2. The KZG PCS defined w.r.t. the bilinear group parameters gp.

4.3 The KZG PCS Family

The construction of the multivariate KZG scheme proposed by Papamanthou, Shi, and Tamassia [PST13] is presented in Figure 2. Compared to [PST13], our presentation relies on our explicit polynomial decomposition lemma, which, e.g., makes it clear that the original univariate PCS by Kate, Zaverucha and Goldberg [KZG10] is, in a very strong sense, a special case of the multivariate variant from [PST13]: The cardinality n of the degree bounds $\mathbf{d}$ decides the number of variables supported by the commitment key ck. Indeed, for a singular degree bound $d \in \mathbb{N}$, the above scheme recovers exactly the original *univariate* KZG scheme [KZG10], where the evaluation proof $\pi = [q(t)]_1$ is constructed using the quotient polynomial $\frac{f(X)-f(x)}{X-x}$, and the verification check is simply $(C - [z]_1) \bullet [1]_2 = \pi \bullet [t - x]_2$ (Figure 10 presents the complete protocol).

Parallel computation of multivariate evaluation proofs. We show that the explicit structure of the quotient polynomials provided by our Lemma 1 can be leveraged through the following observation to improve the practical efficiency of the prover when committing to a multivariate polynomial. Interestingly, it is possible to compute the quotient polynomials more efficiently than what one might expect from the original implicit polynomial decomposition lemma in [PST13]. Specifically, in the non-interactive scheme, the elements of the KZG evaluation proof can be computed in parallel. In [PST13], the quotient polynomials are defined *implicitly* for all $1 \leq i \leq n$ as $r_{i-1}(X_i, \ldots, X_n) = q_i(X_i, \ldots, X_n)(X_i - x_i) + r_i(X_{i+1}, \ldots, X_n)$, where $r_0(X_1, \ldots, X_n) = f(X_1, \ldots, X_n)$. To compute them using the implicit definition, one would divide $f(X_1, \ldots, X_n)$ by $(X_1 - x_1)$ to get the quotient $q_1(X_1, \ldots, X_n)$ and then proceed with dividing the remainder $r_1(X_2, \ldots, X_n)$ etc., which leads to a sequential process. However, by leveraging the explicit structure of q_i's from Lemma 1, if the prover knows the whole evaluation point $(x_1, \ldots, x_n)$ then it can parallelize the computation and directly compute each polynomial $q_i(X_i, \ldots, X_n) = \frac{f(x_1,\ldots,x_{i-1},X_i,\ldots,X_n) - f(x_1,\ldots,x_i,X_{i+1},\ldots,X_n)}{(X_i - x_i)}$, as it only depends on f and $(x_1, \ldots, x_i)$.

Furthermore, in our proof of Lemma 1, we showed that $r_i(X_{i+1}, \ldots, X_n) = f(x_1, \ldots, x_i, X_{i+1}, \ldots, X_n)$ for all i, and we can use this fact to simplify the computation of the quotient polynomials even further. To compute the i-th quotient polynomial, the prover simply computes the quotient in the polynomial division $f(x_1, \ldots, x_{i-1}, X_i, \ldots, X_n)/(X_i - x_i)$ without explicitly computing the remainder. The result is precisely $q_i(X_i, \ldots, X_n)$, since we effectively divide the remainder $r_{i-1}(X_i, \ldots, X_n)$ as required. The key observation is that we already know the explicit form of the remainder, eliminating the need to compute it.

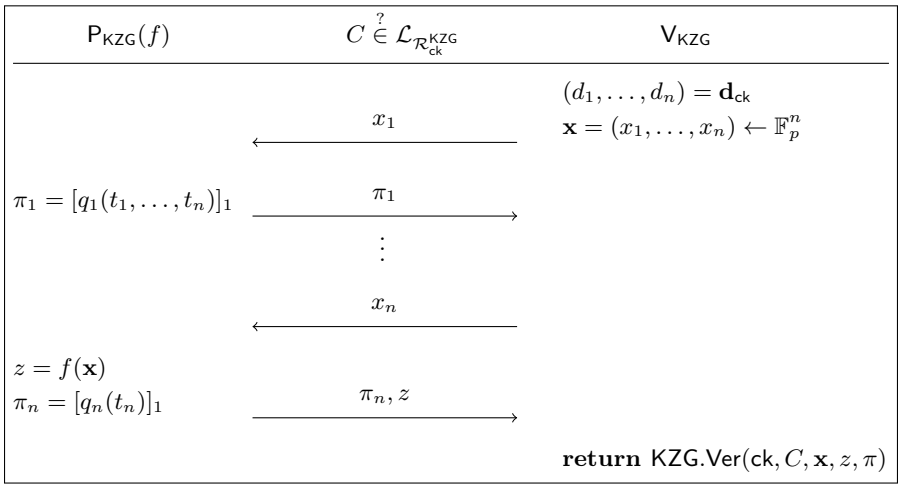

Fig. 3. The extended KZG PoKoP.

4.4 The Canonical and Extended KZG PoKoP

Given a commitment key ck and a commitment C, the canonical PoKoP of the KZG scheme proceeds exactly as the general template presented in Figure 1: the verifier samples a uniform evaluation point $\mathbf{x} = (x_1, \ldots, x_n) \leftarrow \mathbb{F}_q^n$ and sends it to the prover. The prover computes $z = f(x)$ and $\pi = \mathsf{KZG.Open}(\mathsf{ck}, f, \mathbf{x})$ and sends (z, π) to the verifier. The verifier outputs $\mathsf{KZG.Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi)$. Indeed, the KZG polynomial commitment is used in this exact form as a building block in higher-level protocols, such as when compiling PIOPs into SNARKs.

A downside of the above canonical KZG PoKoP is that, in the analysis of its knowledge-soundness, we cannot guarantee any consistency among the evaluation proofs even when the verifier samples challenges $\mathbf{x} = (x_1, \ldots, x_{n-1}, x_n)$ and $\mathbf{x}' = (x_1, \ldots, x_{n-1}, x_n')$ that differ only in the last coordinate. Since for all $i \in [1, n]$ the prover message π_i only depends on $x_1, \ldots, x_i$, we would expect that the first $n-1$ messages of the prover are the same for challenges $\mathbf{x}$ and $\mathbf{x}'$. We show in Section 9 that this is not guaranteed in the canonical KZG PoKoP since there exist many accepting proofs for a single statement and an adversary that obtains one of them can efficiently construct the others.

Next, we present an alternative "extended" PoKoP for the KZG scheme that is easier to analyze while, for all practical purposes, retains the favorable properties of the canonical KZG PoKoP. It is based on the same observation that allows parallel computation of the quotient polynomials as discussed in Section 4.3, that is, each q_i depends only on the first i coordinates of the evaluation point $(x_1, \ldots, x_n)$. The extended KZG PoKoP (Figure 3) proceeds in n rounds. In the i-th round, the verifier sends x_i to the prover, who computes the quotient polynomial q_i and sends back π_i. In the last round, the prover additionally sends $z = f(x_1, \ldots, x_n)$. Given the transcript, the verifier outputs $0/1$ based on the same check as the KZG verifier. Due to the interactive nature of the extended KZG PoKoP, we can apply round-by-round rewinding to it, and, thus, force prefix consistency in transcripts while extracting (see Definition 15).

Definition 10 (Extended KZG PoKoP). *Let* $\mathsf{KZG} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ *be the multivariate polynomial commitment scheme defined in Figure 2. We call the protocol* $(\mathsf{P}_{\mathsf{KZG}}, \mathsf{V}_{\mathsf{KZG}})$ *defined in Figure 3 the* extended KZG PoKoP.

In order to be able to utilize our extended KZG PoKoP in a SNARK compiler [GWC19, LXZ+24, LPS23], we need to transform the interactive proof into a non-interactive proof using the Fiat-Shamir heuristic [FS86]. This means that for all $i \in [1, n]$ we replace the verifier messages x_i by a hash of the previous transcript, i.e., the previous transcript of the SNARK, the commitment C, the $x_1, \ldots, x_{i-1}$-values and the prover messages

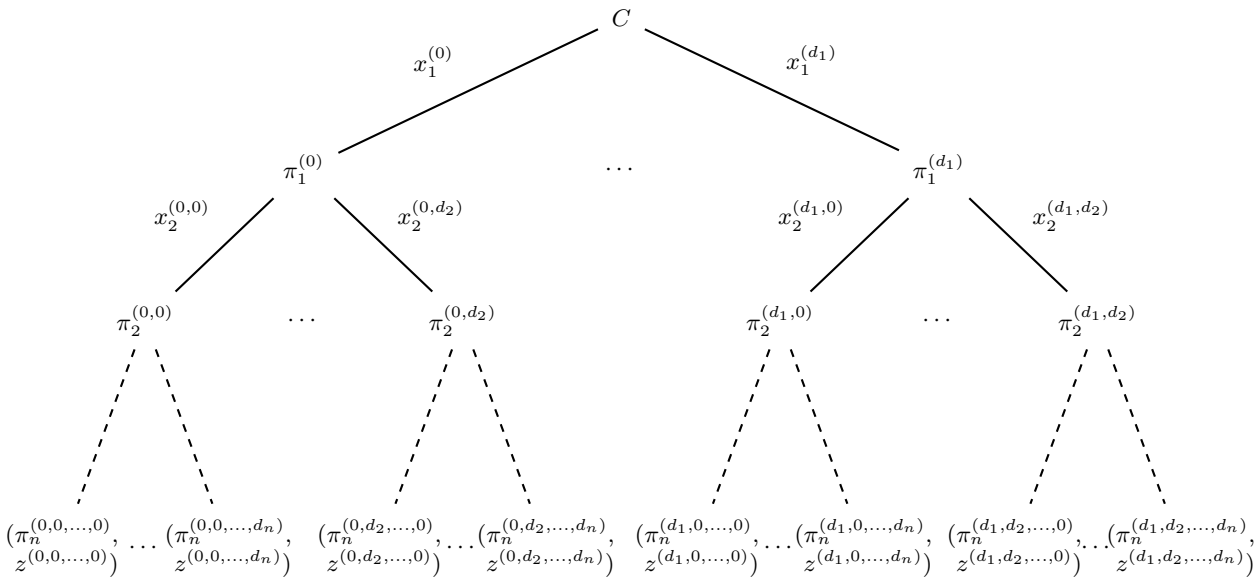

Fig. 4. A $(d_1+1,\ldots,d_n+1)$-tree of accepting transcripts for the extended KZG PoKoP for relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{KZG}}$.

$\pi_1,\ldots,\pi_{i-1}$. Note that this reduces the number of possible evaluation points compared to the canonical KZG PoKoP since $x_2,\ldots,x_n$ are uniquely determined from the input of the hash function. We note that this is not an issue for the SNARK compilers from [GWC19, LXZ+24, LPS23] because there the evaluation points are obtained via the Fiat-Shamir heuristic anyway. Hence, for this application, it is not necessary that the verifier can ask for *any* evaluation point, but it is sufficient that it receives an accepting evaluation proof for a uniformly random evaluation point sampled using the random oracle.

5 Special Soundness of the KZG PoKoPs

In this section, we study the special soundness of the canonical and extended KZG PoKoPs introduced in Section 4.4 as a first step to proving their knowledge-soundness. Under a suitable computational hardness assumption, we show that the natural extraction strategy via interpolation from a tree of accepting transcripts is successful. However, this is easier to argue for the extended KZG PoKoP, which we consider in most of this section. We discuss generalizing our techniques from the extended KZG PoKoP to the canonical one in Section 5.4.

The tree of transcripts for the extended KZG PoKoP. Given a commitment key ck and a commitment C, consider an interaction between a prover trying to convince a verifier that he knows a polynomial f such that $(C,f)\in\mathcal{R}_{\mathsf{ck}}^{\mathsf{KZG}}$ via the extended KZG PoKoP from Figure 3. A $(k_1,\ldots,k_n)$-tree of accepting transcripts $\mathcal{T}$ for this protocol (Definition 4) is simply a structured collection of transcripts for some set of $\prod_{i=1}^n k_i$ distinct evaluation points. Note, that if $\mathbf{d}_{\mathsf{ck}}=(d_1,\ldots,d_n)$ are the degree bounds supported by ck then each $(d_1+1,\ldots,d_n+1)$-tree of accepting transcripts corresponds (via interpolation) to a polynomial that satisfies the degree bounds $\mathbf{d}_{\mathsf{ck}}$ and matches the evaluations recorded in the transcripts. Thus, given a $(d_1+1,\ldots,d_n+1)$-tree of accepting transcripts $\mathcal{T}$, the extractor can simply output the interpolating polynomial corresponding to $\mathcal{T}$.

In Figure 4, we describe the explicit structure of a $(d_1+1,\ldots,d_n+1)$-tree of accepting transcripts $\mathcal{T}$. The root of the tree is labeled by the instance C. The internal vertices are labeled by the prover's messages π_i

and the edges are labeled by the verifier's challenges x_i. The root has exactly d_1+1 children, and, in general, every vertex at level k has exactly d_k+1 children corresponding to different choices for x_k. Every path from the root to a leaf node corresponds to a transcript $(C, x_1, \ldots, x_n, \pi_1, \ldots, \pi_n, z)$. Since the transcript is *accepting*, we have $\mathsf{KZG.Ver}(\mathsf{ck}, C, (x_1, \ldots, x_n), z, (\pi_1, \ldots, \pi_n)) = 1$.

There is a transcript for each leaf node, so we can index the transcripts by vectors of indices $\mathbf{i} = (i_1, \ldots, i_n) \in [\mathbf{d}] = [d_1] \times \cdots \times [d_n]$, where the k-th coordinate i_k of the coordinate vector $\mathbf{i}$ corresponds to the choice of the edge exiting the vertex on the k-th level of the tree. Thus, the tree $\mathcal{T}$ of accepting transcripts corresponds to a set of $\prod_{k=1}^{n}(d_k+1)$ accepting transcripts $\left\{\left(C, x_1^{(\mathbf{i})}, \ldots, x_n^{(\mathbf{i})}, \pi_1^{(\mathbf{i})}, \ldots, \pi_n^{(\mathbf{i})}, z^{(\mathbf{i})}\right)\right\}_{\mathbf{i}\in[\mathbf{d}]}$. Importantly, the tree structure ensures consistency of transcripts sharing prefixes, which we leverage next. We revisit consistency of transcripts in Section 5.4 when extending our techniques to the canonical KZG PoKoP.

5.1 Multivariate Batching Lemma

By interpolating a polynomial from a $(d_1+1, \ldots, d_n+1)$-tree of transcripts, we naturally obtain a polynomial that matches both the degrees and the evaluations recorded in the transcripts. What remains to be argued is that the interpolated polynomial is consistent with the commitment. A crucial step in this process is given in Lemma 2, which enables us to combine the verification checks through the last variable. This results in a single identity that, instead of the evaluations, ensures consistency of a univariate interpolating polynomial $L(X_n)$ to the commitment C. The set of transcripts considered in our n-variate batching lemma is a special subset of transcripts from a $(d_1+1, \ldots, d_n+1)$-tree of transcripts illustrated in Figure 4. Specifically, it corresponds to a path from the root to the n-the level extended by all the leaf nodes, i.e., it is a set of d_n+1 transcripts that share the verifier's challenges $x_1, \ldots, x_{n-1}$ and prover's answers $\pi_1, \ldots, \pi_{n-1}$, while each transcript has a distinct n-the challenge x_n.

Lemma 2 (n-Variate Batching Lemma). *For $n \in \mathbb{N}$ and $t_1, \ldots, t_n \in \mathbb{F}_p$, let*

$$\left\{\left([c]_1, x_1, \ldots, x_{n-1}, x_n^{(i_n)}, [\varphi_1]_1, \ldots, [\varphi_{n-1}]_1, \left([\varphi_n^{(i_n)}]_1, z^{(i_n)}\right)\right)\right\}_{i_n\in[d_n]}$$

be a set of accepting transcripts w.r.t. $t_1, \ldots, t_n$, i.e., such that, for all $i_n \in [d_n]$, it holds that

$$\left[c - z^{(i_n)}\right]_1 \bullet [1]_2 = \left(\bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2\right) \otimes \left[\varphi_n^{(i_n)}\right]_1 \bullet \left[t_n - x_n^{(i_n)}\right]_2. \tag{9}$$

If $\left|\left\{x_n^{(i_n)}\right\}_{i_n=0}^{d_n}\right| = d_n + 1$ *then, for*

$$S = \left\{x_n^{(i_n)}\right\}_{i_n=0}^{d_n}, \quad [\varphi_n]_1 = \sum_{i_n=0}^{d_n} w_{i_n}^S [\varphi_n^{(i_n)}]_1, \quad \text{and} \quad L(X_n) = \sum_{i_n=0}^{d_n} z^{(i_n)} \ell_{i_n}^S(X_n),$$

it holds that $[c - L(t_n)]_1 \bullet [1]_2 = \left(\bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2\right) \otimes [\varphi_n]_1 \bullet [Z_S(t_n)]_2$.

Proof. Since all the elements $x_n^{(i_n)}$ are distinct, the set S is an interpolation domain of cardinality d_n+1 and both $[\varphi_n]_1$ and $L(X_n)$ are well defined. We prove the claimed identity by expanding the left side of the

equation and deriving the expression on the right side as follows:

$$\begin{aligned}
[c - L(t_n)]_1 \bullet [1]_2 &= \left[c - \sum_{i_n=0}^{d_n} z^{(i_n)} \ell_{i_n}^S(t_n)\right]_1 \bullet [1]_2 \\
&= \left[\sum_{i_n=0}^{d_n} \ell_{i_n}^S(t_n)\left(c - z^{(i_n)}\right)\right]_1 \bullet [1]_2 \\
&= \left[\sum_{i_n=0}^{d_n} \ell_{i_n}^S(t_n)\left(\left(\sum_{i=1}^{n-1} \varphi_i(t_i - x_i)\right) + \varphi_n^{(i_n)}\left(t_n - x_n^{(i_n)}\right)\right)\right]_1 \bullet [1]_2 \\
&= \left[\sum_{i=1}^{n-1} \varphi_i(t_i - x_i) + \sum_{i_n=0}^{d_n} Z_S(t_n) w_{i_n}^S \varphi_n^{(i_n)}\right]_1 \bullet [1]_2 \\
&= \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2 \otimes [\varphi_n]_1 \bullet [Z_S(t_n)]_2.
\end{aligned}$$

The correctness of the above derivations can be justified as follows. In the first equality, we used the definition of $L(X_n)$ via Lagrange interpolation. In the second equality, we use the fact that $\sum_{i_n=0}^{d_n} \ell_{i_n}^S(X_n) = 1$. In the third equality, we use eq. (2) from the statement of the lemma, and the bilinearity of the pairing. In the fourth equality, we use the fact that $\ell_{i_n}^S(X)(X - x_n^{(i_n)}) = Z_S(X) w_{i_n}^S$. In the fifth equality, we again used the fact that $\sum_{j=0}^{n} \ell_j^S(Y) = 1$, along with the definition of $[\varphi_n]_1$. Finally, we just moved terms around using the bilinearity of the pairing. □

Our n-variate batching lemma implies as a special case the following *univariate* batching lemma of Lipmaa et al. [LPS24a], which they used in their proof of special soundness of the univariate KZG scheme.

Lemma 3 (Batching Lemma [LPS24a]). *Let $S = \{x_i\}_{i\in[n]} \subseteq \mathbb{F}_p$ with $x_i \neq x_j$ for $i \neq j$. Assume that for all $i \in [n]$:*

$$[c - z_i]_1 \bullet [1]_2 = [\varphi_i]_1 \bullet [\sigma - x_i]_2 \tag{10}$$

for some $[c]_1, [\varphi_i]_1$, σ and z_i. Then

$$[c - L(\sigma)]_1 \bullet [1]_2 = [\varphi]_1 \bullet [Z_S(\sigma)]_2,$$

where $[\varphi]_1 := \sum_{i\in[n]} w_i^S [\varphi_i]_1$ and $L(X) := \sum_{i\in[n]} z_i \ell_i^S(X)$.

The proof of Lemma 3 in [LPS24a] proceeds via a case analysis w.r.t. the trapdoor σ and the evaluation point x, since they leverage the identity $w_i^S = \ell_i^S(X)(X - x)/Z_S(X)$. As a result, their analysis treats the case $\sigma = x$ separately to ensure that w_i^S is defined correctly. In contrast, our proof takes a more direct approach by using the identity $\ell_i^S(X_i)(X_i - x_i) = Z_S(X_i) w_i^S$, eliminating the need to handle the case when $x_i = t_i$ separately.

5.2 Multivariate ARSDH

The univariate batching lemma (Lemma 3) suggests a path towards proving the consistency of the univariate interpolating polynomial $L(X)$ with the commitment $C = [c]_1$, that is, if we could argue $[c - L(\sigma)]_1 = [0]_1$. Note that, without knowing the secret evaluation point σ, it seems hard to produce a nontrivial group element $[c - L(\sigma)]_1$ and some $[\varphi]_1$ that would satisfy the identity $[c - L(\sigma)]_1 \bullet [1]_2 = [\varphi]_1 \bullet [Z_S(\sigma)]_2$. This observation motivates the (univariate) adaptive rational strong Diffie-Hellman (ARSDH) assumption introduced in [LPS24a], which we restate below. Importantly, in our context, the adversary has some control over the interpolation domain, and, thus, the adversary can choose the set S *adaptively* in ARSDH. In this sense, ARSDH is a strengthening of the RSDH assumption from [GR19]. In the univariate ARSDH assumption below, n denotes a univariate degree bound.

Definition 11 (ARSDH Game). *For a* PPT *adversary* $\mathcal{A}$*, a bilinear-group generator* PGen*, and* $n \in \mathsf{poly}(\lambda)$*, the* ARSDH Game *is defined as follows:*

1. *Sample* $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ *and* $\sigma \leftarrow \mathbb{F}_p$.
2. *On input* $1^\lambda, \mathsf{gp}$*, and* $\left(\left[\left(\sigma^i\right)_{i=0}^n\right]_1, [\sigma]_2\right)$*,* $\mathcal{A}$ *outputs* $[\chi]_1$*,* $[\varphi]_1$*, and a set* $S \subseteq \mathbb{F}_p$.

$\mathcal{A}$ *wins if and only if*

$$|S| = n+1, \quad [\chi]_1 \neq [0]_1, \quad \text{and} \quad [\chi]_1 \bullet [1]_2 = [\varphi]_1 \bullet [Z_S(\sigma)]_2.$$

We write $\mathsf{ARSDH.Game}(\mathcal{A}, \mathsf{PGen}, n) = 1$ *if* $\mathcal{A}$ *wins and* 0 *otherwise.*

Definition 12 (ARSDH). *We say the* degree n adaptive rational strong Diffie-Hellman (n-ARSDH) assumption *holds for* PGen *if, for any* PPT $\mathcal{A}$*, it holds:*

$$\Pr[\mathsf{ARSDH.Game}(\mathcal{A}, \mathsf{PGen}, n) = 1] \in \mathsf{negl}(\lambda).$$

Similarly to the univariate batching lemma and ARSDH, our n-variate batching lemma (Lemma 2) motivates a corresponding hardness assumption that can be see as an n-variate generalization of ARSDH.

Definition 13 (ARSDH(n) Game). *For a* PPT *adversary* $\mathcal{A}$*, a bilinear-group generator* PGen*, and* $\mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$*, the* ARSDH($n$) Game *is defined as follows*

1. *Sample* $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ *and* $(t_1, \ldots, t_n) \leftarrow \mathbb{F}_p^n$.
2. *On input* $1^\lambda, \mathsf{gp}$*, and* $\left(\left\{\left[t_1^{i_1} \cdot \ldots \cdot t_n^{i_n}\right]_1\right\}_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, \{[t_i]_2\}_{i\in[1,n]}\right)$*,* $\mathcal{A}$ *outputs*

$$[\chi]_1, \quad \{[\varphi_i]_1\}_{i=1}^{n-1}, \quad \{x_i\}_{i=1}^{n-1} \subseteq \mathbb{F}_p, \quad [\psi]_1, \quad \text{and} \quad S \subseteq \mathbb{F}_p.$$

$\mathcal{A}$ wins if and only if:

- $|S| = d_n + 1$,
- $[\chi]_T \neq \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2$*, and*
- $[\chi]_1 \bullet [1]_2 = \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2 \otimes [\psi]_1 \bullet [Z_S(t_n)]_2$.

We write $\mathsf{ARSDH}(n).\mathsf{Game}(\mathcal{A}, \mathsf{PGen}, \mathbf{d}) = 1$ *if* $\mathcal{A}$ *wins and 0 otherwise.*

Definition 14 (ARSDH(n)). *Let* PGen *be a bilinear group generator. Then the* n-variate Adaptive Rational Strong Diffie-Hellman (ARSDH(n)) assumption *holds for* PGen *if, for any* $\mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$ *and* PPT $\mathcal{A}$*,*

$$\Pr[\mathsf{ARSDH}(n).\mathsf{Game}(\mathcal{A}, \mathsf{PGen}, \mathbf{d}) = 1] \in \mathsf{negl}(\lambda).$$

First note that, when restricted to $n = 1$, i.e., a single variable, ARSDH(1) collapses to the univariate ARSDH from [LPS24a] by the convention that a group product over an empty index set evaluates to the neutral element $[0]_T$ of the group.

Since the winning condition in the n-variate case is more complex, there are various ways to satisfy it trivially. Correspondingly, we need to disallow the adversary from exploiting these, and, thus, we add the second "non-triviality" condition $[\chi]_T \neq \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2$ in Definition 13. This condition is rather general and it rules out various easy ways of satisfying the winning condition, e.g., where the adversary sets $[\chi]_1 = [\varphi_i]_1 = [\psi]_1 = 0$ and choose S and x_i randomly. Additionally, the non-triviality condition naturally arises in the reduction from PDL to ARSDH(n) in the Algebraic Group Model (AGM) (see Section 8), and it allows us to show that ARSDH(n) follows from the $(n, 1)$-PDL assumption in the AGM. This is analogous to the proof of hardness of the univariate ARSDH under $(n, 1)$-PDL in the AGMOS from [LPS24a] that necessitates $[\chi]_1 \neq 0$. In some sense, the AGM analysis supports the choice of the corresponding non-triviality conditions in the definition of their univariate ARSDH (Definition 11) and our multivariate ARSDH (Definition 13).

While $\mathsf{ARSDH}(n)$ is not the most direct generalization of ARSDH, as the vanishing polynomial depends only on the last variable, it leads to a plausibly weaker assumption than the more general variant where the winning condition is based on an identity that involves a vanishing polynomial in each variable. In Section 5.4, we discuss this more general extension of ARSDH to n-variables, which we call GARSDH. Skipping ahead, since GARSDH is a plausibly stronger, it allows for a more direct proof of the special soundness for the multivariate KZG scheme.

$\mathsf{ARSDH}(n)$ implies $\mathsf{ARSDH}(n-1)$. Notice that if we assume $\mathsf{ARSDH}(n)$ then we do not need to additionally assume that $\mathsf{ARSDH}(i)$ holds also for all $i < n$; all the weaker variants are implied by $\mathsf{ARSDH}(n)$. The key observation here is that we can transform an $\mathsf{ARSDH}(n)$ instance with degree bounds $d_1, \ldots, d_n$ into an $\mathsf{ARSDH}(n-1)$ instance with degree bounds $d_2, \ldots, d_n$ by setting $[\varphi_1]_1 = [0]_1$, effectively eliminating the first variable.

Lemma 4. *For all $n \in \mathsf{poly}(\lambda)$, $\mathsf{ARSDH}(n)$ implies $\mathsf{ARSDH}(n-1)$.*

Proof. Assume we have $\left(\left\{\left[t_1^{i_1} \cdot \ldots \cdot t_n^{i_n}\right]_1\right\}_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, \{[t_i]_2\}_{i\in[1,n]}\right)$ given to us as input for the $\mathsf{ARSDH}(n)$ challenge. The idea is we can give the oracle for $\mathsf{ARSDH}(n-1)$ a version of our input where we ignore the first variable and then use the output as our answer along with setting $[\varphi_1]_1 = [0]_1$.

We run the $\mathsf{ARSDH}(n-1)$ oracle on input $\left(\left\{\left[t_2^{i_2} \cdot \ldots \cdot t_n^{i_n}\right]_1\right\}_{i_2,\ldots,i_n=0}^{d_2,\ldots,d_n}, \{[t_i]_2\}_{i\in[2,n]}\right)$, where the degrees $(d_2, \ldots, d_n)$ remain in $\mathsf{poly}(\lambda)$.

The oracle gives us $[\chi']_1, [\varphi_i']_1, [\psi']_1, S', x_i'$ for $i \in [1, n-1]$ such that

- $|S'| = d_n + 1$,
- $[\chi']_T \neq \bigotimes_{i=1}^{n-2} [\varphi_i']_1 \bullet [t_{i+1} - x_i']_2$,
- $[\chi']_1 \bullet [1]_2 = \bigotimes_{i=1}^{n-2} [\varphi_i']_1 \bullet [t_{i+1} - x_i']_2 \otimes [\psi']_1 \bullet [Z_{S'}(t_n)]_2$.

If we set $[\chi]_1 = [\chi']_1, [\varphi_i]_1 = [\varphi_{i-1}'], [\psi]_1 = [\psi']_1, S = S', x_i = x_{i-1}'$ for $i \in [2, n-1]$ and $[\varphi_1]_1 = [0]_1, x_1 \leftarrow \mathbb{F}_p$ we have

- $|S| = d_n + 1$,
- $[\chi]_T \neq \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2$,
- $[\chi]_1 \bullet [1]_2 = \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2 \otimes [\psi]_1 \bullet [Z_S(t_n)]_2$,

where all of these are straightforward from the properties of the elements from the oracle. □

5.3 Special Soundness of the Extended KZG PoKoP

In this section, we use $\mathsf{ARSDH}(n)$ (Definition 12) and our multivariate batching lemma (Lemma 2) to prove the special soundness of the extended KZG PoKoP from Figure 3 as stated in the following theorem.

Theorem 4. *Let $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ and $\mathsf{ck} \leftarrow \mathsf{KZG.KGen}(1^\lambda, (d_1, \ldots, d_n), \mathsf{gp})$ for some $d_1, \ldots, d_n \in \mathbb{N}$. If the $\mathsf{ARSDH}(n)$ assumption holds for PGen, then the extended KZG PoKoP presented in Figure 3 is a $(d_1 + 1, \ldots, d_n + 1)$-special sound interactive argument for the relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{KZG}}$.*

In the proof, we inductively consolidate the KZG verification equations using our n-variate batching lemma (Lemma 2), progressively reducing them until we obtain a single equation that establishes the consistency of the interpolating polynomial with the commitment. We can view this process through the tree representation of the transcripts in Figure 4. We begin with consistency at the level of leaves, each corresponding to some value $z^{(\mathbf{i})}$ at an evaluation point $\mathbf{x}^{(\mathbf{i})}$. Moving up the tree, we analyze partial interpolating polynomials, where each step is defined by fixing a prefix of the evaluation point. At each step i, we apply the $\mathsf{ARSDH}(n-i)$ assumption to guarantee that the partial interpolating polynomial remains consistent with the commitment. This process continues until we reach the root of the tree, at which point we establish the final consistency condition for the full interpolating polynomial and the commitment.

$\mathsf{Ext}_{\mathsf{ck}}(C, \mathcal{T})$:

1. parse $\mathcal{T}$ as $\left\{\left(C, x_1^{(\mathbf{i})}, \ldots, x_n^{(\mathbf{i})}, \left[\varphi_1^{(\mathbf{i})}\right]_1, \ldots, \left(\left[\varphi_n^{(\mathbf{i})}\right]_1, z^{(\mathbf{i})}\right)\right)\right\}_{\mathbf{i}\in[\mathbf{d}]}$
2. for all $k \in [1, n]$, set $S_k^{(i_1,\ldots,i_{k-1})} = \left\{x_k^{(i_1,\ldots,i_k)}\right\}_{i_k=0}^{d_k}$
3. **return** $L(X_1, \ldots, X_n) = \sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} z^{(i_1,\ldots,i_n)} \prod_{k=1}^{n} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(X_k)$

Fig. 5. Interpolating extractor establishing special soundness for the extended KZG PoKoP.

Importantly, our proof relies solely on the structure of the verification check and not on the specifics of computation of the proof. Thus, it would apply to any scheme based on KZG that utilizes a check of the form $[c - z]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n} [\varphi_j]_1 \bullet [t_j - x_j]_2$, and allows for extracting a tree of transcripts compatible with our batching lemma.

Proof (Theorem 4). Let $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ and $\mathsf{ck} \leftarrow \mathsf{KZG.KGen}(1^\lambda, (d_1, \ldots, d_n), \mathsf{gp})$ for some $d_1, \ldots, d_n \in \mathbb{N}$ as in the statement of the theorem. The commitment key ck specifies the commitment relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{KZG}}$.

We prove the theorem by analyzing the interpolating extractor $\mathsf{Ext}_{\mathsf{ck}}$ defined in Figure 5. On input $C \in \mathbb{G}_1$ and a $(d_1 + 1, \ldots, d_n + 1)$-tree of accepting transcripts $\mathcal{T}$ for the extended KZG PoKoP produced by some PPT algorithm Tree, the extractor uses $\mathbf{i} = (i_1, \ldots, i_n) \in [\mathbf{d}] = [d_1] \times \cdots \times [d_n]$ to index the leaves and parses $\mathcal{T}$ as a set of accepting transcripts

$$\left\{\left(C, x_1^{(\mathbf{i})}, \ldots, x_n^{(\mathbf{i})}, \left[\varphi_1^{(\mathbf{i})}\right]_1, \ldots, \left(\left[\varphi_n^{(\mathbf{i})}\right]_1, z^{(\mathbf{i})}\right)\right)\right\}_{\mathbf{i}\in[\mathbf{d}]}.$$

Since $\mathcal{T}$ has the tree structure depicted in Figure 4, the transcripts are consistent on shared prefixes, and, thus, we can denote $x_j^{(\mathbf{i})} = x_j^{(i_1,\ldots,i_n)} = x_j^{(i_1,\ldots,i_j)}$ and $\left[\varphi_j^{(\mathbf{i})}\right]_1 = \left[\varphi_j^{(i_1,\ldots,i_n)}\right]_1 = \left[\varphi_j^{(i_1,\ldots,i_j)}\right]_1$ for all $\mathbf{i} \in [\mathbf{d}]$ and $j \in [1, n]$.

For all $k \in [1, n]$, the extractor defines interpolation subdomains $S_k^{(i_1,\ldots,i_{k-1})} = \left\{x_k^{(i_1,\ldots,i_k)}\right\}_{i_k=0}^{d_k}$. Then, it uses the corresponding values $z^{(\mathbf{i})}$ to construct and output an interpolating polynomial

$$L(X_1, \ldots, X_n) = \sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} z^{(i_1,\ldots,i_n)} \prod_{k=1}^{n} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(X_k).$$

By the structure of the tree and the condition that each node at level k must have $d_k + 1$ edges with distinct labels connecting it to level $k + 1$, the leaves of $\mathcal{T}$ correspond to $\prod_{i\in[1,n]}(d_i + 1)$ distinct evaluation points. Moreover, for each prefix of indices $(i_1, \ldots, i_{k-1})$, the interpolation subdomain $S_k^{(i_1,\ldots,i_{k-1})}$ is of size $d_k + 1$. Thus, the polynomial $L(X_1, \ldots, X_n)$ satisfies the degree bounds, i.e., $\deg_{X_i}(L) \leq d_i$ for all $i \in [1, n]$, and matches all the evaluations specified by the tree, i.e., $L(x_1^{(\mathbf{i})}, \ldots, x_n^{(\mathbf{i})}) = z^{(\mathbf{i})}$ for all $\mathbf{i} \in [\mathbf{d}]$.

To show that $(C, L(X_1, \ldots, X_n)) \in \mathcal{R}_{\mathsf{ck}}^{\mathsf{KZG}}$, it remains to be argued that, with an overwhelming probability, $[c - L(t_1, \ldots, t_n)]_1 = [0]_1$ when $\mathcal{T}$ was produced by some PPT algorithm Tree on input C and $(d_1+1, \ldots, d_n+1)$. We show this holds under the $\mathsf{ARSDH}(n)$ assumption.

Since Tree produces a tree of accepting transcripts, we have that, for all $(i_1, \ldots, i_n) \in [\mathbf{d}]$, it holds that

$$\left[c - z^{(i_1,\ldots,i_n)}\right]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n} \left[\varphi_j^{(i_1,\ldots,i_j)}\right]_1 \bullet \left[t_j - x_j^{(i_1,\ldots,i_j)}\right]_2.$$

Using Lemma 2, we get that, for all $(i_1, \ldots, i_{n-1}) \in [d_1] \times \cdots \times [d_{n-1}]$,

$$\left[c - g^{(i_1,\ldots,i_{n-1})}(t_n)\right]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n-1} \left[\varphi_j^{(i_1,\ldots,i_j)}\right]_1 \bullet \left[t_j - x_j^{(i_1,\ldots,i_j)}\right]_2 \otimes \left[\varphi_n^{(i_1,\ldots,i_{n-1})}\right]_1 \bullet \left[Z_{S_n^{(i_1,\ldots,i_{n-1})}}(t_n)\right]_2, \quad (11)$$

where we define $\left[\varphi_n^{(i_1,\ldots,i_{n-1})}\right]_1 = \sum_{i_n=0}^{d_n} w_{i_n}^{S_n^{(i_1,\ldots,i_{n-1})}} \left[\varphi_n^{(i_1,\ldots,i_n)}\right]_1$ and the partially interpolated polynomials $g^{(i_1,\ldots,i_{n-1})}(X_n) = \sum_{i_n=0}^{d_n} z^{(\mathbf{i})} \ell_{i_n}^{S_n^{(i_1,\ldots,i_{n-1})}}(X_n)$. Now, assuming that $\mathsf{ARSDH}(n)$ holds, we have that, except with a negligible probability,

$$\left[c - g^{(i_1,\ldots,i_{n-1})}(t_n)\right]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n-1} \left[\varphi_j^{(i_1,\ldots,i_j)}\right]_1 \bullet \left[t_j - x_j^{(i_1,\ldots,i_j)}\right]_2 \tag{12}$$

for all $(i_1,\ldots,i_{n-1}) \in [d_1] \times \cdots \times [d_{n-1}]$. Otherwise, we would be able to break ARSDH by setting $[\chi]_1 = \left[c - g^{(i_1,\ldots,i_{n-1})}(t_n)\right]_1$, $[\varphi_j]_1 = \left[\varphi_j^{(i_1,\ldots,i_j)}\right]_1$, $[\psi]_1 = \left[\varphi_n^{(i_1,\ldots,i_{n-1})}\right]_1$ and $S = S_n^{(i_1,\ldots,i_{n-1})}$, where $(i_1,\ldots,i_{n-1})$ is the index for which eq. (12) does not hold.

Thus, we can assume eq. (12) holds with overwhelming probability for all $(i_1,\ldots,i_{n-1}) \in [d_1]\times\cdots\times[d_{n-1}]$, and we can again use Lemma 2 on these equalities, giving us

$$\left[c - h^{(i_1,\ldots,i_{n-2})}(t_{n-1},t_n)\right]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n-2} \left[\varphi_j^{(i_1,\ldots,i_j)}\right]_1 \bullet \left[t_j - x_j^{(i_1,\ldots,i_j)}\right]_2 \otimes \left[\varphi_{n-1}^{(i_1,\ldots,i_{n-2})}\right]_1 \bullet \left[Z_{S_{n-1}^{(i_1,\ldots,i_{n-2})}}(t_{n-1})\right]_2,$$

where

$$\left[\varphi_{n-1}^{(i_1,\ldots,i_{n-2})}\right]_1 = \sum_{i_{n-1}=0}^{d_{n-1}} w_{i_{n-1}}^{S_{n-1}^{(i_1,\ldots,i_{n-2})}} \left[\varphi_{n-1}^{(i_1,\ldots,i_{n-1})}\right]_1,$$

and

$$h^{(i_1,\ldots,i_{n-2})}(X_{n-1},X_n) = \sum_{i_{n-1}=0}^{d_{n-1}} g^{(i_1,\ldots,i_{n-1})}(X_n) \ell_{i_{n-1}}^{S_{n-1}^{(i_1,\ldots,i_{n-2})}}(X_n).$$

By Lemma 4, we can assume that $\mathsf{ARSDH}(n-1)$ holds, and therefore, for all $(i_1,\ldots,i_{n-2}) \in [d_1]\times\cdots\times[d_{n-2}]$, we have that, except with a negligible probability,

$$\left[c - h^{(i_1,\ldots,i_{n-2})}(t_{n-1},t_n)\right]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n-2} \left[\varphi_j^{(i_1,\ldots,i_j)}\right]_1 \bullet \left[t_j - x_j^{(i_1,\ldots,i_j)}\right]_2,$$

giving us enough equations to proceed to the next variable using Lemma 2.

In this way, we continue to reduce the statement by leveraging assumptions $\mathsf{ARSDH}(n-2),\ldots,\mathsf{ARSDH}(1)$, ultimately demonstrating that if $[c - L(t_1,\ldots,t_n)]_1 \neq [0]_1$ with a non-negligible probability, we would be able to break one of these variants of ARSDH at some intermediate step. However, as noted in Lemma 4, $\mathsf{ARSDH}(n)$ implies $\mathsf{ARSDH}(n-1)$ for all $n \in \mathsf{poly}(\lambda)$. Therefore, the theorem holds assuming $\mathsf{ARSDH}(n)$. □

5.4 General Interpolating Extractor

In this section, we examine the necessary modifications to the definitions in Section 5.2 and 5.1 to accommodate a more general interpolation approach that would allow us to handle the *canonical* KZG PoKoP.

First, let us revisit the discussion on the structure of the transcripts in the extended PoKoP from Section 5. There, we directly obtain the specific properties that we need to apply our n-variate batching lemma due to the tree structure of the transcripts. The canonical KZG PoKoP has the *single round* structure depicted in the Figure 1. As a result, in our interpolation approach towards the proof of special soundness, we would start with a $(\prod_{i\in[1,n]}(d_i+1))$-tree of accepting transcripts that has only a *single level* and can be viewed as a set of transcripts

$$\left\{\left(C, (x_1^{(\mathbf{i})},\ldots,x_n^{(\mathbf{i})}), ((\pi_1^{(\mathbf{i})},\ldots,\pi_n^{(\mathbf{i})}), z^{(\mathbf{i})})\right)\right\}_{\mathbf{i}\in[\mathbf{d}]}.$$

Note that, it is not of particular importance that the transcripts are tuples of tuples. Hence, we consider each transcript only as a single tuple of all the elements appearing in it, i.e., $\left(C, x_1^{(\mathbf{i})},\ldots,x_n^{(\mathbf{i})}, \pi_1^{(\mathbf{i})},\ldots,\pi_n^{(\mathbf{i})}, z^{(\mathbf{i})}\right)$,

to simplify the notation. Moreover, this allows us to talk simultaneously about the consistency properties of the canonical and extended KZG PoKoP. Next, we define the consistency properties.

Definition 15 (Consistency and π-Consistency). *For $\mathcal{I} \subseteq [\mathbf{d}]$, we say that a set of transcripts*

$$\left\{\left(C, x_1^{(\mathbf{i})}, \ldots, x_n^{(\mathbf{i})}, \pi_1^{(\mathbf{i})}, \ldots, \pi_n^{(\mathbf{i})}, z^{(\mathbf{i})}\right)\right\}_{\mathbf{i}\in\mathcal{I}}$$

is consistent *if the following properties hold:*

1. ***Shared commitment:*** *All transcripts are w.r.t. the same C.*
2. ***Consistency on shared prefixes:*** *For all $\mathbf{i}, \mathbf{j} \in \mathcal{I}$ with the structure $\mathbf{i} = (i_1, \ldots, i_k, i_{k+1}, \ldots, i_n)$ and $\mathbf{j} = (i_1, \ldots, i_k, j_{k+1}, \ldots, j_n)$ for some $k \in [1, n]$, it holds*

$$\left(x_1^{(\mathbf{i})}, \ldots, x_k^{(\mathbf{i})}\right) = \left(x_1^{(\mathbf{j})}, \ldots, x_k^{(\mathbf{j})}\right).$$

3. ***Distinctness of evaluation points:*** *For all $\mathbf{i}, \mathbf{j} \in \mathcal{I}$, if $\mathbf{i} \neq \mathbf{j}$ then*

$$\mathbf{x}^{\mathbf{i}} = \left(x_1^{(\mathbf{i})}, \ldots, x_n^{(\mathbf{i})}\right) \neq \mathbf{x}^{\mathbf{j}} = \left(x_1^{(\mathbf{j})}, \ldots, x_n^{(\mathbf{j})}\right).$$

We call a set of transcript π-consistent *if it is consistent and, additionally, the following holds:*

4. ***Consistency on shared prefixes for proofs:*** *For all $\mathbf{i}, \mathbf{j} \in \mathcal{I}$ such that $\mathbf{i} = (i_1, \ldots, i_k, i_{k+1}, \ldots, i_n)$ and $\mathbf{j} = (i_1, \ldots, i_k, j_{k+1}, \ldots, j_n)$ for some $k \in [1, n]$, it holds that*

$$\left(\pi_1^{(\mathbf{i})}, \ldots, \pi_k^{(\mathbf{i})}\right) = \left(\pi_1^{(\mathbf{j})}, \ldots, \pi_k^{(\mathbf{j})}\right).$$

Clearly, all the transcripts of both the extended and canonical PoKoP share the same root C. Consistency on shared prefixes states formally that any pair of transcripts corresponding to a pair of paths sharing a prefix up to level $k \in [n]$ can differ only after the k-th message of the prover. The distinctness of evaluation points ensures that the set of evaluation points $\{\mathbf{x}^{\mathbf{i}}\}_{\mathbf{i}\in[\mathbf{d}]}$ corresponding to the whole tree defines a feasible interpolation domain.

Optimally, we would like to have consistency or even π-consistency for such a set of transcripts. However, unlike for a $(d_1 + 1, \ldots, d_n + 1)$-tree of transcripts for the extended KZG PoKoP, neither consistency nor π-consistency follows directly from the structure of an arbitrary $(\prod_{i\in[1,n]}(d_i + 1))$-tree of accepting transcripts for the canonical KZG PoKoP. We deal with this nuisance as follows: 1) we prove a generalized multivariate batching lemma that does not necessitate π-consistent transcripts and allows working only with consistent transcripts instead, and 2) in section 6, we show that it is possible to efficiently extract a *consistent* $(\prod_{i\in[1,n]}(d_i + 1))$-tree of accepting transcripts for the canonical KZG PoKoP. Importantly, by avoiding π-consistency, there is no need for round-by-round rewinding when extracting consistent transcripts, making it possible to establish knowledge-soundness for the canonical KZG PoKoP. Finally, we note that, even though we establish computational special soundness of the canonical KZG PoKoP only relative to a more restricted class of *consistent* trees produced by some PPT algorithm, it is not an issue as our ultimate goal is proving knowledge-soundness and special soundness serves only as a useful intermediate notion aiding the modularity of our proof.

Generalized multivariate batching lemma. Our *generalized* multivariate batching lemma consolidates *all* equalities from a set of $(\prod_{i\in[1,n]}(d_i + 1))$ consistent accepting transcripts into a single statement concerning the interpolated polynomial $L(X_1, \ldots, X_n)$. However, this approach makes the resulting equality more complex as each pairing now incorporates a vanishing polynomial.

Lemma 5 (Generalized n-variate Batching Lemma). *For $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{N}^n$ and $(t_1, \ldots, t_n) \in \mathbb{F}_p^n$, let*

$$\left\{ \left([c]_1, (x_1^{(\mathbf{i})}, \ldots, x_n^{(\mathbf{i})}), ((\pi_1^{(\mathbf{i})}, \ldots, \pi_n^{(\mathbf{i})}), z^{(\mathbf{i})}) \right) \right\}_{\mathbf{i} \in [\mathbf{d}]}$$

be a set of consistent transcripts according to Definition 15 that are all accepting w.r.t. $(t_1, \ldots, t_n)$, i.e., such that, for all $\mathbf{i} \in [\mathbf{d}]$, it holds that

$$\left[c - z^{(\mathbf{i})}\right]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n} \left[\varphi_j^{(\mathbf{i})}\right]_1 \bullet \left[t_j - x_j^{(\mathbf{i})}\right]_2 .$$

Denote $x_j^{(i_1,\ldots,i_n)} = x_j^{(i_1,\ldots,i_j)}$ (due to prefix consistency) and, for all $j \in [1, n]$ and $(i_1, \ldots, i_{j-1}) \in [d_1] \times \cdots \times [d_{j-1}]$, define:

- $S_j^{(i_1,\ldots,i_{j-1})} = \left\{ x_j^{(i_1,\ldots,i_j)} \right\}_{i_j=0}^{d_j}$,
- $\left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 = \sum_{i_j,\ldots,i_n=0}^{d_j,\ldots,d_n} \prod_{k \neq j} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(t_k) w_{i_j}^{S_j^{(i_1,\ldots,i_{j-1})}} \left[\varphi_j^{(i_1,\ldots,i_n)}\right]_1$, *and*
- $L(X_1, \ldots, X_n) = \sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} z^{(i_1,\ldots,i_n)} \prod_{k=1}^{n} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(X_k)$.

Then, it holds that

$$[c - L(t_1, \ldots, t_n)]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2 .$$

Proof. The proof is analogous to the proof of the n-variate batching lemma (Lemma 2). Using the same observations, we can expand the left side of the claimed equality and derive the right side as follows:

$$\begin{aligned}
&[c - L(t_1, \ldots, t_n)]_1 \bullet [1]_2 = \\
&= \left[c - \sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} z^{(i_1,\ldots,i_n)} \prod_{k=1}^{n} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(t_k) \right]_1 \bullet [1]_2 \\
&= \left[\sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} \prod_{k=1}^{n} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(t_k)(c - z^{(i_1,\ldots,i_n)}) \right]_1 \bullet [1]_2 \\
&= \left[\sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} \prod_{k=1}^{n} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(t_k) \left(\sum_{j=1}^{n} \varphi_j^{(i_1,\ldots,i_n)} \left(t_j - x_j^{(i_1,\ldots,i_n)} \right) \right) \right]_1 \bullet [1]_2 \\
&= \left[\sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} \sum_{j=1}^{n} \prod_{k=1}^{n} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(t_k) \varphi_j^{(i_1,\ldots,i_n)} (t_j - x_j^{(i_1,\ldots,i_n)}) \right]_1 \bullet [1]_2 \\
&= \left[\sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} \sum_{j=1}^{n} \prod_{k \neq j} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(t_k) \varphi_j^{(i_1,\ldots,i_n)} Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j) w_{i_j}^{S_j^{(i_1,\ldots,i_{j-1})}} \right]_1 \bullet [1]_2 \\
&= \left[\sum_{j=1}^{n} \sum_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \varphi_j^{(i_1,\ldots,i_{j-1})} Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j) \right]_1 \bullet [1]_2 \\
&= \bigotimes_{j=1}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2 ,
\end{aligned}$$

which proves the claim. □

Generalized multivariate ARSDH assumption. Our generalized multivariate batching lemma (Lemma 5) naturally leads to a generalization of the ARSDH assumption we define below.

Definition 16 (GARSDH Game). *For a* PPT *adversary* $\mathcal{A}$*, a bilinear-group generator* PGen*, and* $\mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$*, the* GARSDH($n$) Game *is defined as follows:*

1. *Obtain* $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ *and sample* $t_1, \ldots, t_n \leftarrow \mathbb{F}_p$.
2. *On input* $1^\lambda, \mathsf{gp}$ *and* $\left(\left\{\left[t_1^{i_1} \cdot \ldots \cdot t_n^{i_n}\right]_1\right\}_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, \{[t_i]_2\}_{i\in[1,n]}\right)$ $\mathcal{A}$ *outputs for all* $j \in [1,n]$ *and for all* $(i_1, \ldots, i_{j-1}) \in [d_1] \times \cdots \times [d_{j-1}]$:
   - $[\chi]_1$,
   - $\left\{S_j^{(i_1,\ldots,i_{j-1})}\right\}_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}}$, *where* $S_j^{(i_1,\ldots,i_{j-1})} \subseteq \mathbb{F}_p$,
   - $\left\{\left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1\right\}_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}}$.

$\mathcal{A}$ *wins if and only if:*

- $|S_j^{(i_1,\ldots,i_{j-1})}| = d_j + 1$ *for all* $j \in [1,n]$ *and all* $i_1 \in [d_1], \ldots, i_{j-1} \in [d_{j-1}]$,
- $[\chi]_T \neq \bigotimes_{j=2}^n \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$, *and*
- $[\chi]_T = \bigotimes_{j=1}^n \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$.

We write $\mathsf{GARSDH}(n).\mathsf{Game}(\mathcal{A}, \mathsf{PGen}, \mathbf{d}) = 1$ *if* $\mathcal{A}$ *wins and 0 otherwise.*

Definition 17 (GARSDH(n)). *Let* PGen *be a bilinear group generator. Then, the* n-variate Generalized Adaptive Rational Strong Diffie-Hellman (GARSDH(n)) assumption *holds for* PGen *if, for all* $\mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$ *and* PPT $\mathcal{A}$*, it holds that*

$$\Pr[\mathsf{GARSDH.Game}(\mathcal{A}, \mathsf{PGen}, \mathbf{d}) = 1] \in \mathsf{negl}(\lambda).$$

Recall from our discussion on ARSDH in Section 5.2 that the second condition in its security game ensures the adversary cannot win trivially. This is the case also for the second condition in Definition 16 of the GARSDH security game, which takes care of various trivial ways of satisfying the last winning condition, e.g, where the adversary use a solution for GARSDH(1) and sets all other elements to 0 to win. Similarly to ARSDH, the non-triviality condition is, to some extent, motivated by our analysis of GARSDH in the AGM (see Section 8).

As for ARSDH, we can show that GARSDH(n) reduces to GARSDH for a smaller number of variables. The proof is even more straightforward than for ARSDH, as there is no need to shift the proofs of the instances with a smaller number of variables to satisfy the second condition.

Lemma 6. *For all* $n \in \mathsf{poly}(\lambda)$*,* GARSDH($n$) *implies* GARSDH($n-1$).

Proof. Assume we have $\left(\left\{\left[t_1^{i_1} \cdot \ldots \cdot t_n^{i_n}\right]_1\right\}_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, \{[t_i]_2\}_{i\in[1,n]}\right)$ given to us as input for the GARSDH(n) challenge. The idea is we can give the oracle for GARSDH($n-1$) a version of our input where we ignore the last variable and then use the output as our answer along with setting $[\varphi_n]_1 = [0]_1$.

We run the GARSDH($n-1$) oracle on input $\left(\left\{\left[t_1^{i_1} \cdot \ldots \cdot t_{n-1}^{i_{n-1}}\right]_1\right\}_{i_1,\ldots,i_{n-1}=0}^{d_1,\ldots,d_{n-1}}, \{[t_i]_2\}_{i\in[1,n-1]}\right)$, where $(d_1, \ldots, d_{n-1})$ remains in $\mathsf{poly}(\lambda)$.

The oracle gives us for all $j \in [1, n-1]$: $[\chi]_1, \{S_j^{(i_1,\ldots,i_{j-1})}\}_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}}$, $\left\{\left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1\right\}_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}}$, such that

$\mathsf{Ext}_{\mathsf{ck}}(C, \mathcal{T})$:

1. parse $\mathcal{T}$ as $\left\{\left(C, \left(x_1^{(i_1)}, \ldots, x_n^{(i_1,\ldots,i_n)}\right), \left(\left([\varphi_1^{(\mathbf{i})}]_1, \ldots, [\varphi_n^{(\mathbf{i})}]_1\right), z^{(\mathbf{i})}\right)\right)\right\}_{(\mathbf{i})\in[\mathbf{d}]}$
2. for all $k \in [1, n]$, set $S_k^{(i_1,\ldots,i_{k-1})} = \left\{x_k^{(i_1,\ldots,i_k)}\right\}_{i_k=0}^{d_k}$
3. **return** $L(X_1, \ldots, X_n) = \sum_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n} z^{(i_1,\ldots,i_n)} \prod_{k=1}^{n} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(X_k)$

Fig. 6. Interpolating extractor establishing special soundness for the canonical KZG PoKoP.

1. $|S_j^{(i_1,\ldots,i_{j-1})}| = d_j + 1$ for all $j \in [1, n-1]$ and all $i_1 \in [d_1], \ldots, i_{j-1} \in [d_{j-1}]$,
2. $[\chi]_T \neq \bigotimes_{j=2}^{n-1} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$,
3. $[\chi]_T = \bigotimes_{j=1}^{n-1} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$.

Set $[\varphi_n]_1 = [0]_1$, all $\{S_n^{(i_1,\ldots,i_{n-1})}\}_{i_1,\ldots,i_{n-1}=0}^{d_1,\ldots,d_{n-1}}$ to $d_n + 1$ distinct random elements of $\mathbb{F}_p$ and output the rest of the things exactly as from the oracle. This gives us

1. $|S_j^{(i_1,\ldots,i_{j-1})}| = d_j + 1$ for all $j \in [1, n]$ and all $i_1 \in [d_1], \ldots, i_{j-1} \in [d_{j-1}]$,
2. $[\chi]_T \neq \bigotimes_{j=2}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$,
3. $[\chi]_T = \bigotimes_{j=1}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$,

where all of these are straightforward from the properties of the elements from the oracle. □

Special soundness of the canonical KZG PoKoP under GARSDH. The special soundness of the canonical KZG PoKoP holds under the GARSDH assumption, as stated in the theorem below.

Theorem 5. *Let* $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ *and* $\mathsf{ck} \leftarrow \mathsf{KZG.KGen}(1^\lambda, (d_1, \ldots, d_n), \mathsf{gp})$ *for some* $d_1, \ldots, d_n \in \mathbb{N}$. *If the* $\mathsf{GARSDH}(n)$ *assumption holds for* PGen *then the canonical KZG* PoKoP *is a computationally* $(\prod_{i=1}^{n}(d_i+1))$*-special sound interactive argument for the relation* $\mathcal{R}_{\mathsf{ck}}^{\mathsf{KZG}}$ *w.r.t. the class of tree-constructing algorithms that output consistent trees (Definition 15).*

Proof. Let $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ and $\mathsf{ck} \leftarrow \mathsf{KZG.KGen}(1^\lambda, (d_1, \ldots, d_n), \mathsf{gp})$ for some $d_1, \ldots, d_n \in \mathbb{N}$ as in the statement of the theorem. The commitment key ck specifies the commitment relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{KZG}}$.

Similarly to the proof of special soundness for the extended KZG PoKoP (Theorem 4), we prove the theorem by analyzing the interpolating extractor $\mathsf{Ext}_{\mathsf{ck}}$ defined in Figure 6. The input to the extractor is a commitment $C \in \mathbb{G}_1$ and a $(\prod_{i\in[1,n]}(d_i + 1))$-tree of *consistent* accepting transcripts $\mathcal{T}$ for the canonical KZG PoKoP. Note that consistency is an additional requirement that we must impose here, as it does not naturally follow from the tree structure of $\mathcal{T}$, unlike in the case of the extended KZG PoKoP.

The main difference from the extractor in Theorem 4 (Figure 5) is that a tree of accepting transcripts for the canonical KZG PoKoP contains transcripts of the form

$$\left\{\left([c]_1, (x_1^{(i_1)}, x_2^{(i_1,i_2)}, \ldots, x_n^{(i_1,\ldots,i_n)}), (([\varphi_1^{(\mathbf{i})}]_1, \ldots, [\varphi_n^{(\mathbf{i})}]_1), z^{(\mathbf{i})})\right)\right\}_{\mathbf{i}\in[\mathbf{d}]},$$

i.e., there is a single challenge from the verifier and an answer from the prover. Importantly, by the consistency of the transcripts (Definition 15), we can still argue that the leaves of $\mathcal{T}$ correspond to $\prod_{i\in[1,n]}(d_i+1)$ distinct evaluation points. Moreover, for each prefix of indices $(i_1, \ldots, i_{k-1})$, the interpolation subdomain $S_k^{(i_1,\ldots,i_{k-1})}$

is of size d_k+1. Thus, the polynomial $L(X_1,\ldots,X_n)$ output by the extractor satisfies the degree bounds, i.e., $\deg_{X_i}(L) \leq d_i$ for all $i \in [1,n]$, and matches all evaluations specified by the tree, i.e., $L(x_1^{(\mathbf{i})},\ldots,x_n^{(\mathbf{i})}) = z^{(\mathbf{i})}$ for all $\mathbf{i} \in [\mathbf{d}]$.

It remains to be argued that, except with a negligible probability, the interpolated polynomial $L(X_1,\ldots,X_n)$ matches the commitment, i.e., that $[c - L(t_1,\ldots,t_n)]_1 = [0]_1$. To this end, we use the generalized multivariate batching lemma and $\mathsf{GARSDH}(n)$. Because the transcripts are accepting, we have that, for all $(i_1,\ldots,i_n) \in [\mathbf{d}]$,

$$\left[c - z^{(i_1,\ldots,i_n)}\right]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n} \left[\varphi_j^{(i_1,\ldots,i_n)}\right]_1 \bullet \left[t_j - x_j^{(i_1,\ldots,i_j)}\right]_2 .$$

Using Lemma 5, we get

$$[c - L(t_1,\ldots,t_n)]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2 ,$$

where

$$\left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 = \sum_{i_j,\ldots,i_n=0}^{d_j,\ldots,d_n} \prod_{k\neq j} \ell_{i_k}^{S_k^{(i_1,\ldots,i_{k-1})}}(t_k) w_{i_j}^{S_j^{(i_1,\ldots,i_{j-1})}} \left[\varphi_j^{(i_1,\ldots,i_n)}\right]_1 .$$

Assuming that $\mathsf{GARSDH}(n)$ holds, we get that, except with a negligible probability,

$$[c - L(t_1,\ldots,t_n)]_1 \bullet [1]_2 = \bigotimes_{j=2}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2 \tag{13}$$

for all $(i_1,\ldots,i_{n-1}) \in [d_1] \times \cdots \times [d_{n-1}]$, as otherwise we would be able to break $\mathsf{GARSDH}(n)$ by setting $[\chi]_1 = [c - L(t_1,\ldots,t_n)]_1$.

Assuming that $\mathsf{GARSDH}(n-1)$ holds, eq. (13) implies, except with a negligible probability, the equality

$$[c - L(t_1,\ldots,t_n)]_1 \bullet [1]_2 = \bigotimes_{j=3}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2 ,$$

as, otherwise, we would be able to break $\mathsf{GARSDH}(n-1)$ in a similar way as before.

In this way, we can continue to reduce the statement by leveraging $\mathsf{GARSDH}(n-2),\ldots,\mathsf{GARSDH}(1)$, ultimately demonstrating that if $[c-L(t_1,\ldots,t_n)]_1 \neq [0]_1$, we would be able to break one of these variants of GARSDH. However, as noted in Lemma 6, $\mathsf{GARSDH}(n)$ implies $\mathsf{GARSDH}(n-1)$ for all $n \in \mathsf{poly}(\lambda)$. Therefore, the only assumption required is $\mathsf{GARSDH}(n)$. □

6 Knowledge-Soundness of the KZG PoKoPs

In this section, we recall a general rewinding strategy by Attema, Cramer, and Kohl [ACK21] that extracts an accepting tree of transcripts for a public-coin protocol with polynomial number of rounds in expected polynomial time. Then, we use the ACK extraction strategy to prove the knowledge-soundness of the two KZG PoKoPs under ARSDH, respectively GARSDH.

6.1 The [ACK21] Tree-Finding Game and Algorithm

[ACK21] presented the following abstract game that distils the task of extracting a tree of accepting transcripts given black-box rewinding access to a prover.

Algorithm 1 The tree-finding algorithm TREE from [ACK21].

$\text{TREE}^H(k_1, \ldots, k_n)$:
1: $a \leftarrow R$
2: **return** $\text{TREE}_0^H(a)$ with the root labeled by a

$\text{TREE}_n^H(a, x_1, \ldots, x_n)$:
1: **if** $H[a, x_1, \ldots, x_n] = 1$ **then**
2: **return** New leaf labeled by $H[a, x_1, \ldots, x_n]$
3: **else**
4: **return** $\perp$

$\text{TREE}_i^H(a, x_1, \ldots, x_i)$: ▷ for $i \in \{0, \ldots, n-1\}$
1: $x_{i+1} \leftarrow N$
2: v a new vertex
3: $\mathcal{T} = \text{TREE}_{i+1}^H(a, x_1, \ldots, x_{i+1})$
4: **if** $\mathcal{T} = \perp$ **then**
5: **return** $\perp$
6: **else**
7: Connect v and the root of $\mathcal{T}$ by an edge labeled by x_{i+1}
8: **while** v has $< k_i$ children **do**
9: $x_{i+1} \leftarrow N$ (without replacement)
10: $\mathcal{T} = \text{TREE}_{i+1}^H(a, x_1, \ldots, x_{i+1})$
11: **if** $\mathcal{T} \neq \perp$ **then**
12: Connect v and the root of $\mathcal{T}$ by an edge labeled by x_{i+1}
13: **if** All possibilities were tried **then**
14: **return** $\perp$

The tree-finding game. For parameters $n, R, N \in \mathbb{N}$, let $H \in \{0,1\}^{R \times N \times \cdots \times N}$ be an $(n+1)$-dimensional 0/1-tensor. Given query access to H and a tuple $(k_1, \ldots, k_n) \in [N]^n$, our goal is to find a $(k_1, \ldots, k_n)$-*tree of 1-entries in* H satisfying the following structure:

1. The root r is labeled by $a \in [R]$.
2. A vertex v in the i-th level (the root r is in the first level) has k_i children. The edges connecting v and its children are labelled by k_i distinct values from $[N]$.
3. Let v be a leaf and $x_1, \ldots, x_n$ the edge-labels on the path from the root r to v. Then, $H[a, x_1, \ldots, x_n] = 1$.

Any $(k_1, \ldots, k_n)$-tree of 1-entries in H is basically a *structured* set of locations in H containing 1-entries. As we explain next, an algorithm that succeeds in the tree-finding game can be used to extract a $(k_1, \ldots, k_n)$-tree of accepting transcripts in an n-round public-coin interactive protocol given rewinding black-box access to a prover. Indeed, [ACK21] used the tree-finding game in their proof of a general forking lemma for multi-round protocols.

For an n-round protocol (P, V), set R as the size of the universe of P's randomness and N as the size of the universe of V's challenges. The $(n+1)$-dimensional tensor $H \in \{0,1\}^{R \times N \times \cdots \times N}$ is then defined such that, for any fixed P's randomness $a \in [R]$ and vector of V's challenges $x_1, \ldots, x_n \in [N]$, the entry $H[r, x_1, \ldots, x_n]$ equals 1 if and only if V eventually accepts based on P's responses computed on challenges $x_1, \ldots, x_n$ using the fixed P's randomness a. Note that the 1-entries in H correspond exactly to accepting protocol transcripts, and, thus, any $(k_1, \ldots, k_n)$-tree of 1-entries in H gives a $(k_1, \ldots, k_n)$-tree of accepting transcripts for (P, V).

For example for the extended KZG PoKoP $(\mathsf{P}_{\mathsf{KZG}}, \mathsf{V}_{\mathsf{KZG}})$, the number of rounds n is the number of variables supported by the commitment key ck, $R = 2^{\mathsf{poly}(\lambda)}$, and $N = |\mathbb{F}_p|$. The entries of the corresponding $(n+1)$-dimensional tensor H_{KZG} are defined by the outputs of $\mathsf{V}_{\mathsf{KZG}}$, and we wish to find a $(d_1 + 1, \ldots, d_n + 1)$-tree of 1-entries in H_{KZG}.

The tree-finding algorithm. Attema et al. [ACK21] also proposed a tree-finding algorithm $\textsc{Tree}$ presented in Algorithm 1. Their algorithm $\textsc{Tree}$ first samples an entry in H to estimate the density of 1-entries in H. If it sampled a 1-entry then it keeps sampling without replacement until it either produces a $(k_1, \ldots, k_n)$-tree of 1-entries of H or queries the whole H. In case the initial query was a 0-entry, the algorithm terminates. Importantly, they proved the following strong guarantees on the performance of $\textsc{Tree}$.

Lemma 7 ([ACK21]). *For $n, R, N \in \mathbb{N}$, let $H \in \{0,1\}^{R \times N \times \cdots \times N}$ be an $(n+1)$-dimensional tensor and let $\varepsilon \in \mathbb{R}$ be the fraction of 1-entries in H. Then, $\textsc{Tree}^H(k_1, \ldots, k_n)$ outputs a $(k_1, \ldots, k_n)$-tree of 1-entries in H with probability at least*

$$\varepsilon - \frac{\sum_{i=1}^{n}(k_i - 1)}{N},$$

and the expected sample complexity of $\textsc{Tree}^H(k_1, \ldots, k_n)$ is at most $\prod_{i=1}^{n} k_i$.

Probability of discarding sampled 1-entries. Next, we argue that the algorithm $\textsc{Tree}$ is very likely to add all the sampled 1-entries from H to the constructed tree. We start with the following basic observation.

Observation 6 *For all $H \in \{0,1\}^{R \times N \times \cdots \times N}$ and $(k_1, \ldots, k_n) \in [N]^n$, the algorithm $\textsc{Tree}^H(k_1, \ldots, k_n)$ makes at least N queries before discarding any 1-entry it sampled from H.*

Proof. Suppose we are in the middle of execution of the procedure $\textsc{Tree}_i$. The procedure has already successfully called the procedure $\textsc{Tree}_{i+1}$ at least once, i.e., it found some 1-entries of H. Thus, the while cycle beginning at Line 8 is executing. Then, the procedure $\textsc{Tree}_i$ aborts only at Line 14 after trying all possibilities for the challenge x_{i+1}. Thus, it had to call the procedure $\textsc{Tree}_{i+1}$ N times and it follows the procedure made at least N queries to H. □

Lemma 7 and Observation 6 imply that, if successful, the algorithm $\textsc{Tree}$ discards any found 1-entry only with a small probability whenever the size of the $(k_1, \ldots, k_n)$-tree is smaller than N.

Corollary 1. *For all $H \in \{0,1\}^{R \times N \times \cdots \times N}$ and $(k_1, \ldots, k_n) \in [N]^n$, if the algorithm $\textsc{Tree}^H(k_1, \ldots, k_n)$ succeeded then it sampled and discarded a 1-entry with probability at most $(\prod_{i=1}^{n} k_i)/N$.*

Proof. Let Q be the random variable counting the number of queries made by $\textsc{Tree}^H(k_1, \ldots, k_n)$ and D be the event that $\textsc{Tree}^H(k_1, \ldots, k_n)$ discarded a sampled 1-entry. By Lemma 7, we have that $\mathrm{E}[Q] \leq \prod_{i=1}^{n} k_i$. By Observation 6, we have that $\mathrm{E}[Q \mid D] \geq N$. Therefore, we get that

$$\prod_{i=1}^{n} k_i \geq \mathrm{E}[Q] = \Pr[D] \cdot \mathrm{E}[Q \mid D] + \Pr[\neg D] \cdot \mathrm{E}[Q \mid \neg D] \geq \Pr[D] \cdot \mathrm{E}[Q|D] \geq \Pr[D] \cdot N,$$

where the second inequality holds since Q is non-negative. Note that we get the statement of the corollary after dividing the above inequality by N. □

6.2 Knowledge-Soundness of the KZG PoKoPs

Using the [ACK21] rewinding strategy described in the previous section, we can prove the knowledge-soundness of the two KZG PoKoPs as stated in the following theorem.

Theorem 7. *For $n \in \mathbb{N}$, let $\mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$ be a vector of degree bounds and PGen be a bilinear group generator.*

1. *If the $\mathsf{ARSDH}(n)$ assumption holds for PGen then the extended KZG PoKoP is a proof of knowledge of a polynomial for KZG with respect to PGen and $\mathbf{d}$.*
2. *If the $\mathsf{GARSDH}(n)$ assumption holds for PGen then both extended and canonical KZG PoKoPs are proofs of knowledge of a polynomial for KZG with respect to PGen and $\mathbf{d}$.*

Proof. By Theorem 4, we know that the extended KZG PoKoP is computationally $(d_1+1,\ldots,d_n+1)$-special sound under the $\mathsf{ARSDH}(n)$ assumption. Similarly, Theorem 5 gives that the canonical KZG PoKoP is computationally $((d_1+1)\cdots(d_n+1))$-special sound under $\mathsf{GARSDH}(n)$ with respect to PPT algorithms that output *consistent* $((d_1+1)\cdots(d_n+1))$-trees of accepting transcripts (w.r.t. Definition 15 of consistency). To prove knowledge-soundness of the two protocols, it remains to construct an expected polynomial time algorithm that, given black-box rewinding access to any prover P^* that on input C convinces the verifier $\mathsf{V}_{\mathsf{KZG}}$ with probability δ, outputs a (consistent) tree of accepting transcripts, of the respective size, with probability at least $\delta - \mathsf{negl}(\lambda)$.

We construct such an algorithm via the ACK rewinding approach presented in Section 6.1. We define an $(n+1)$-dimensional 0/1 tensor $H \in \{0,1\}^{R\times p\times\ldots\times p}$ as follows: The entry $H_{i_0,i_1,\ldots,i_n}$ contains the output of $\mathsf{KZG.Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi)$, where i_0 denotes the randomness a of P^* (which determines z and π) and $i_j = x_j \in \mathbb{F}_p$ for all $j \in [1,n]$. Note that we can emulate an oracle for H by interacting with and (if needed) rewinding P^* as follows. For the extended KZG PoKoP, we run P^* on input C with challenges $x_1,\ldots,x_n$ to obtain $H_{a,x_1,\ldots,x_n}$ for some $a \in R$ sampled by P^*. In order to obtain, e.g., the element $H_{a,x_1,\ldots,x_{n-1},x'_n}$, we rewind the adversary to after its $(n-1)$-th message and then send the new challenge x'_n. Using this rewinding technique, we can emulate access to H making sure the randomness a, as well as all shared prefixes of P^*'s messages remain fixed. For the canonical KZG PoKoP, we do not need to ensure consistency of prover's answers. We can just rewind the prover P^* to the outset of the protocol and perform a new run for each evaluation point.

We denote by $\textsc{Tree}^{\mathsf{P}^*}$ the algorithm that follows Algorithm 1 while given emulated access to the oracle H by rewinding P^* as explained above. By Lemma 7, $\textsc{Tree}^{\mathsf{P}^*}(d_1+1,\ldots,d_n+1)$ outputs a $(d_1+1,\ldots,d_n+1)$-tree of 1-entrees in H with probability at least $\delta - \sum_{i=1}^n d_i/p$ and expected sample complexity at most $\prod_{i=1}^n (d_i+1)$. Since the degree bounds d_i are polynomial in λ and the size p of the field is exponential in λ, the success probability of $\textsc{Tree}^{\mathsf{P}^*}$ is at least $\delta - \mathsf{negl}(\lambda)$ and the expected sample complexity is polynomial in λ. Note that, by rewinding P^* in the extended KZG PoKoP while emulating H and running $\textsc{Tree}^{\mathsf{P}^*}$, we implicitly construct a $(d_1+1,\ldots,d_n+1)$-tree of accepting transcripts and the knowledge-soundness of the extended KZG PoKoP follows. Similarly, by rewinding P^* in the canonical KZG PoKoP while emulating H and running $\textsc{Tree}^{\mathsf{P}^*}$, we implicitly construct a $(\prod_{i=1}^n (d_i+1))$-tree of accepting transcripts. Morever, this tree is *consistent*, as it corresponds to a tree of 1-entries in the $(n+1)$-dimensional tensor H, which ensures prefix consistency of the evaluation points. Thus, the knowledge-soundness of the canonical KZG PoKoP follows. □

6.3 KZG PoKoPs vs. Black-Box Extractability in [LPS23]

In this section, we first compare our notion of PoKoP for a PCS to the notion of black-box extractability from [LPS23], and then show that our extractor satisfies also their definition.

Lipmaa, Parisella and Siim's [LPS23, LPS24a] definition of black-box extractability for univariate polynomial commitment schemes can be stated for multivariate polynomial commitment schemes as follows. For a polynomial commitment scheme $\mathsf{PCS} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$, a commitment key ck, and a transcript $\mathsf{tr} = (C, \mathbf{x}, z, \pi)$ define the relation

$$\mathcal{R}^{\mathsf{PCS}}_{\mathsf{ck},\mathsf{tr}} = \{(C,f) \mid C = \mathsf{PCS.Com}(\mathsf{ck}, f) \wedge \overline{\deg}(f) \le \mathbf{d}_{\mathsf{ck}} \wedge f(\mathbf{x}) = z\}.$$

Note that, compared to the commitment relation

$$\mathcal{R}^{\mathsf{PCS}}_{\mathsf{ck}} = \left\{(C,f) \mid C = \mathsf{PCS.Com}(\mathsf{ck}, f) \wedge \overline{\deg}(f) \le \mathbf{d}_{\mathsf{ck}}\right\}$$

we use to define the knowledge-soundness property for PoKoPs, the relation $\mathcal{R}^{\mathsf{PCS}}_{\mathsf{ck},\mathsf{tr}}$ additionally requires the polynomial f to be consistent with the evaluation proof of an initial transcript tr (but not necessarily with the proof π contained in tr).

Definition 18 (Black-box Extractability [LPS23]). *A polynomial commitment scheme* PCS *is* black-box extractable *for* PGen*, if there exists an expected* PPT *black-box extractor* Ext_{BB}*, such that for any*

$n, \mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$ *and all* $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$*, where* $\mathcal{A}_0$ *is* PPT *and* $\mathcal{A}_1$ *is* DPT*, it holds that*

$$\Pr\left[\begin{array}{l}\mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi) = 1 \\ \wedge\, (C, f) \notin \mathcal{R}^{\mathsf{PCS}}_{\mathsf{ck},\mathsf{tr}}\end{array}\middle|\begin{array}{l}\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda), \\ \mathsf{ck} \leftarrow \mathsf{KGen}(\mathsf{gp}, \mathbf{d}), \\ (C, \mathsf{st}) \leftarrow \mathcal{A}_0(\mathsf{ck}), \\ \mathbf{x} \leftarrow \mathbb{F}^n, \\ (\pi, z) \leftarrow \mathcal{A}_1(\mathsf{st}, \mathbf{x}), \\ \mathsf{tr} \leftarrow (C, \mathbf{x}, z, \pi), \\ f \leftarrow \mathsf{Ext}_{BB}^{\mathcal{A}_1(\mathsf{st},\cdot)}(\mathsf{ck}, \mathsf{tr})\end{array}\right] \in \mathsf{negl}(\lambda).$$

To some extent, the experiment captured by the above definition implicitly considers the canonical PoKoP of PCS as defined by us but requires it to be an argument of knowledge for the relation $\mathcal{R}^{\mathsf{PCS}}_{\mathsf{ck},\mathsf{tr}}$. Importantly, by considering the event $\mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi) = 1 \wedge (C, f) \notin \mathcal{R}^{\mathsf{PCS}}_{\mathsf{ck},\mathsf{tr}}$, the polynomial output by $\mathsf{Ext}_{\mathrm{BB}}^{\mathcal{A}_1(\mathsf{st},\cdot)}$ must, except with a negligible probability, match the value z at point $\mathbf{x}$ as specified by the transcript tr. We wish to point out that, despite the similarities, our definition of extractability via PoKoPs seems more conducive to generalizations of the techniques from [LPS24a]. For example, by decoupling the canonical PCS PoKoP construction from the general notion of PoKoP, we were able to invoke the extraction strategy from [ACK21] as a black box, whereas [LPS24a] had to adapt the [ACK21] extraction procedure to their definition and reprove its properties already for the univariate KZG scheme.

Next, we argue that our analysis of the canonical and extended KZG PoKoP*s* demonstrates that the corresponding interactive evaluation proofs satisfy the above definition of black-box extractability. To this end, we show that we can adapt the extractor in the proof of Theorem 7 to output a polynomial f that is consistent with $(\mathbf{x}, z)$ in the initial transcript tr output by $\mathcal{A}_1$. For the canonical KZG PoKoP, this follows directly from Corollary 1. We can just view $\mathbf{x}$ as the first evaluation point on which the algorithm $\textsc{Tree}^{\mathcal{A}_1}$ in the proof of Theorem 7 queries $\mathcal{A}_1$. If π is an accepting proof then, by Corollary 1, the probability that $\mathbf{x}$ is not part of the final tree of transcripts, i.e., inconsistent with extractor's interpolating polynomial f, is negligible.

For the extended KZG PoKoP, it is not immediately clear that the above holds since here we need a proof π for $\mathbf{x}$ that is consistent with the rest of the proofs in the tree. Note, however, that there is no need to keep the proof π output by $\mathcal{A}_1$ as part of the tree. The algorithm $\textsc{Tree}^{\mathcal{A}_1}$ can also query $\mathcal{A}_1$ again on the same point $\mathbf{x}$ and if $\mathcal{A}_1$ outputs (z', π') then start rewinding the adversary exactly as in the proof of Theorem 7. Since the multivariate KZG scheme is evaluation binding on average under the SBDH assumption (see Section 10), we know that if the proof π' is accepting then $z' = z$. If $\mathcal{A}_1$ does not output an accepting proof, we keep querying $\mathcal{A}_1$ on $\mathbf{x}$ until we get an accepting proof. It remains to be argued that, with high probability, this procedure terminates in expected polynomial time. Let Q denote the expected number of queries we make on $\mathbf{x}$. Equivalently, the probability that $\mathcal{A}_1$ outputs an accepting proof is $1/Q$. Since $\mathcal{A}_1$ already managed to output one accepting proof for $\mathbf{x}$, we can assume that the probability of $\mathcal{A}_1$ outputting an accepting proof for $\mathbf{x}$ is inversely polynomial, and, hence, Q is a polynomial.

7 The Extractability of Other KZG Variants

We have established both the special soundness and knowledge-soundness of the basic multivariate KZG from [PST13]. Notably, our proof of special soundness relies solely on the structure of the verification check, while extending to knowledge-soundness requires only the ability to efficiently extract a tree of accepting, consistent transcripts. As a result, our techniques apply not only to KZG but also to other KZG-like schemes that share these basic structural properties. In this section, we consider some extensions of the KZG scheme from the literature and discuss the applicability of our results to them.

$\mathsf{bKZG.KGen}(1^\lambda, \mathbf{d} = (M-1, T-1), \mathsf{aux} = \mathsf{gp} = (p, [1]_1, [1]_2, \bullet))$:

Sample $\tau_X, \tau_Y \leftarrow \mathbb{F}_p$,

Output $\mathsf{ck} = \left(\mathsf{gp}, [\tau_X]_1, [\tau_Y]_1, [\tau_X]_2, [\tau_Y]_2, (U_{i,j})_{i,j=0}^{M-1,T-1} = ([R_i(\tau_Y)L_j(\tau_X)]_1)_{i,j=0}^{M-1,T-1}\right)$.

$\mathsf{bKZG.Com}(\mathsf{ck}, f)$:

1. Parse $f(Y,X)$ as $\sum_{i=0}^{M-1}\sum_{j=0}^{T-1} f_{i,j}L_j(X)R_i(Y)$, and denote $f_i(X) = \sum_{j=0}^{T-1} f_{i,j}L_j(X)$.
2. For all $i \in [M-1]$, compute $\mathsf{com}_{f_i} = \sum_{j=0}^{T-1} f_{i,j}U_{i,j}$.
3. Then combine as $\mathsf{com}_f = \sum_{i=0}^{M-1} \mathsf{com}_{f_i}$.

Output C.

$\mathsf{bKZG.Open}(\mathsf{ck}, f, (y,x))$:

1. For all $i \in [M-1]$ compute
   - $f_i(x)$,
   - $q_1^{(i)}(X) = \frac{f_i(X)-f_i(x)}{X-x}$,
   - $\pi_1^{(i)} = [R_i(\tau_Y)q_1^{(i)}(\tau_X)]_1$.
   - $\pi_1 = \sum_{i=0}^{M-1} \pi_1^{(i)}$
2. Combine as
   - $f(Y,x) = \sum_{i=0}^{M-1} R_i(Y)f_i(x)$
   - $z = f(y,x)$
   - $q_2(Y) = \frac{f(Y,x)-f(y,x)}{Y-y}$
   - $\pi_2 = [q_2(\tau_Y)]_1$
   - $\pi = (\pi_1, \pi_2)$

Output $(z, \pi = (\pi_1, \pi_2))$.

$\mathsf{bKZG.Ver}(\mathsf{ck}, C, (y,x), z, \pi = (\pi_1, \pi_2), \mathsf{ck})$:

If $(C - [z]_1) \bullet [1]_2 = \pi_1 \bullet [\tau_X - x]_2 \otimes \pi_2 \bullet [\tau_Y - y]_2$, output 1, and 0 otherwise.

Fig. 7. The Pianist variant of the bivariate KZG scheme [LXZ+24]

Handling commitment keys for alternative polynomial bases. So far, we have considered the case where $\mathsf{KGen}(1^\lambda, \mathbf{d} = (d_1, \ldots, d_n), \mathsf{gp})$ outputs a commitment key ck w.r.t. a standard monomial basis, i.e., $\mathsf{ck} = \left(\mathsf{gp}, \left([t_1^{i_1} \cdots t_n^{i_n}]_1\right)_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, ([t_i]_2)_{i=1}^n\right)$. Note, however, that the KGen algorithm can output the commitment key ck with respect to an arbitrary polynomial basis such as the Lagrange basis (see Definition 1). Then ck is of the following form

$$\left(\mathsf{gp}, \left([l_{i_1}^{S_1}(t_1) \cdots l_{i_n}^{S_n}(t_n)]_1\right)_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, ([t_i]_2)_{i=1}^n\right),$$

where, for all $j \in [1,n]$, $\{l_{i_j}^{S_j}(X_j)\}_{i_j \in [d_j]}$ is the Lagrange basis with respect to the variable X_j and some interpolation domain $S_j \subseteq \mathbb{F}_p$. Working with the above commitment key constructed w.r.t. the Lagrange basis is beneficial, for example, when the prover wants to commit to a polynomial f represented via evaluations on the interpolation domain, as it saves one application of the inverse Fourier transform.

Importantly, these constructions of ck are equivalent, in the sense that a commitment key expressed in the Lagrange basis can be efficiently transformed into one in the standard monomial basis, and vice versa. For instance, Fast Fourier Transform (FFT) can be used to convert the commitment key to the monomial basis, ensuring efficiency and correctness in the transformation.

7.1 Distributed Bivariate KZG from Pianist [LXZ+24]

Pianist [LXZ+24] is a distributed zero-knowledge proof system that, instead of relying on a monolithic prover, allows the splitting of proof generation among multiple weaker machines with minimal communication overhead, improving proving scalability. In this subsection, we describe how our extractability results apply to the distributed bivariate KZG (DKZG) scheme introduced in [LXZ+24], which is a basic building block in the Pianist protocol.

The DKZG scheme. We present the scheme in Figure 7. The notation is slightly different from our notation in previous sections, as we strive to follow closely the notation used in [LXZ+24]. Still, we keep the additive bracket notation for source groups. The main idea of DKZG is to split the bivariate polynomial $f(Y, X)$ into multiple polynomials to parallelize the computation by introducing sub-provers. Each sub-prover P_i has a univariate polynomial $f_i(X)$ on which they compute locally and simultaneously with the other sub-provers. When all of the sub-provers finished their part of the computation, the master prover P_0 aggregates the partial computations into one and sends it to the verifier.

The way the master prover splits the polynomial is quite natural. Suppose the polynomial $f(Y, X)$ is of the form $\sum_{i=0}^{M-1}\sum_{j=0}^{T-1} f_{i,j}L_j(X)R_i(Y)$, where $L_j(X)$ is the Lagrange basis with respect to the variable X and $R_i(Y)$ is the Lagrange basis with respect to variable Y. Specifically, for $i \in [M-1]$ and $j \in [T-1]$, define

$$L_j(X) = \frac{\omega_X^j}{T}\frac{X^T - 1}{X - \omega_X^j}, \quad R_i(Y) = \frac{\omega_Y^i}{M}\frac{Y^M - 1}{Y - \omega_Y^i},$$

where ω_X and ω_Y are some appropriate primitive roots of unity defining the interpolation domain for X and Y.

The distribution of the bivariate polynomial $f(Y, X)$ is achieved using the following decomposition

$$f(Y, X) = \sum_{i=0}^{M-1} R_i(Y)\left(\sum_{j=0}^{T-1} f_{i,j}L_j(X)\right) = \sum_{i=0}^{M-1} R_i(Y)f_i(X),$$

where $f_i(X) = \sum_{j=0}^{T-1} f_{i,j}L_j(X)$ is the polynomial received by sub-prover P_i. In the scheme, the master prover P_0 performs aggregation exploiting that it just needs to aggregate the values based on the sub-polynomials $f_i(X)$ with respect to the variable Y.

Naturally, the commitment key ck in DKZG contains group elements $([R_i(\tau_Y)L_j(\tau_X)]_1)_{i,j=0}^{M-1,T-1}$, where $T-1, M-1$ are the degrees of variables X and Y respectively, along with all of the other necessary information. In the commitment phase, each sub-prover computes the commitment com_{f_i} to its polynomial $f_i(X)$ and sends it to P_0. The master prover P_0 computes the aggregated commitment $C = \sum_{i=0}^{M-1} com_{f_i}$ and sends it to the verifier.

To compute an evaluation proof for $f(Y, X)$ at a point (y, x), each sub-prover P_i computes $f_i(x)$ and $\pi_1^{(i)}$, which is a "univariate KZG commitment" to the quotient polynomial $q_1^{(i)}(X) = \frac{f_i(X)-f_i(x)}{X-x}$ offset by $R_i(\tau_Y)$. Each sup-prover P_i then sends $f_i(x)$ and $\pi_1^{(i)}$ to P_0. The master prover P_0 aggregates all partial proofs $\pi_1^{(i)}$ into $\pi_1 = \sum_{i=0}^{M-1}\pi_1^{(i)}$. Then, P_0 computes $f(Y, x) = \sum_{i=0}^{M-1} R_i(Y)f_i(x)$, the evaluation of $f(y, x)$, and the quotient polynomial $q_2(Y) = \frac{f(Y,x)-f(y,x)}{Y-y}$. It then constructs π_2 as a commitment to $q_2(Y)$. Finally, P_0 sends $(z, \pi = (\pi_1, \pi_2))$ to the verifier, who accepts based on the verification check

$$(C - [z]_1) \bullet [1]_2 = \pi_1 \bullet [\tau_X - x]_2 \otimes \pi_2 \bullet [\tau_Y - y]_2,$$

which is exactly the bivariate KZG check.

Extractability of DKZG under ARSDH and GARSDH. To illustrate the assumption underlying the proof of extractability for the bivariate DKZG scheme, we present the ARSDH(2) security game explicitly in Definition 19 below.

Definition 19 (Bivariate ARSDH Game). *For a* PPT *adversary* $\mathcal{A}$*, a bilinear-group generator* PGen*, and* $n, m \in \mathsf{poly}(\lambda)$*, the* Bivariate ARSDH Game *is defined as follows:*

1. *Sample* $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ *and* $\sigma, \tau \leftarrow \mathbb{F}_p$.
2. *On input* $1^\lambda, \mathsf{gp}$ *and* $\left(\left(\left[\sigma^i\tau^j\right]_1\right)_{i,j=0}^{m,n}, [\sigma]_2, [\tau]_2\right)$ $\mathcal{A}$ *outputs* $[\chi]_1, [\varphi]_1, [\psi]_1$*,* $\alpha \in \mathbb{F}_p$*, and a set* $S \subseteq \mathbb{F}_p$.

$\mathcal{A}$ wins if and only if:

$$|S| = m+1, \quad \textit{and} \quad [\chi]_T \neq [\varphi]_1 \bullet [\sigma - \alpha]_2, \quad \textit{and} \quad [\chi]_1 \bullet [1]_2 = [\varphi]_1 \bullet [\sigma-\alpha]_2 \otimes [\psi]_1 \bullet [Z_S(\tau)]_2.$$

We write $\mathsf{ARSDH.Game}(\mathcal{A}, \mathsf{PGen}, n, m) = 1$ *if $\mathcal{A}$ wins and 0 otherwise.*

First note that, even though the scheme is defined using a commitment key w.r.t. the Lagrange basis, we can still reduce to ARSDH or GARSDH postulated w.r.t. to the standard monomial basis since it is possible to efficiently compute the DKZG commitment from $\left([\tau^i\sigma^j]_1\right)_{i,j=0}^{m,n}$ and the rest of the input in the ARSDH, resp. GARSDH game. Moreover, the DKZG scheme still supports the extended PoKoP, as all sub-provers only need the X-coordinate of the evaluation point (x, y) to construct the sub-proofs $\pi_1^{(i)}$. Thus, we can extract π-consistent transcripts and rely on ARSDH when analyzing the extended DKZG PoKoP. Additionally, as we already pointed out, the verification check is exactly that of the standard bivariate KZG. Thus, as a direct corollary of our Theorem 7, we get extractability for the extended DKZG PoKoP under ARSDH(2) and for the canonical DKZG PoKoP under GARSDH(2).

Theorem 8. *Let $\mathbf{d} = (M-1, T-1) \in \mathsf{poly}(\lambda)$ be degree bounds and* PGen *be a bilinear group generator.*

1. *If the* ARSDH(2) *assumption holds for* PGen *then the extended* DKZG PoKoP *is a proof of knowledge of a polynomial for* DKZG *with respect to* PGen *and* $\mathbf{d}$.
2. *If the* GARSDH(2) *assumption holds for* PGen *then both the extended and canonical* DKZG PoKoP*s are proofs of knowledge of a polynomial for* DKZG *with respect to* PGen *and* $\mathbf{d}$.

7.2 Distributed Multivariate KZG from HyperPianist [LLZ+24]

HyperPianist is a distributed zero-knowledge proof system designed to achieve linear-time prover cost and logarithmic communication cost. Similarly to how the Pianist ZKP [LXZ+24] improves the scalability of proving in the Plonk proof system [GWC19], the HyperPianist scheme extends the HyperPlonk multivariate proof system [CBBZ23] by efficiently distributing computation across multiple machines with minimal additional overhead. In this section, we show that our extractability results extend to the distributed multilinear KZG (deMKZG) scheme introduced in [LLZ+24] to serve as a building block in their scheme.

We present the deMKZG scheme in Figure 8, following closely the notation from their paper. The deMKZG builds on top of the basic multilinear KZG scheme [PST13]. The authors observe that, since an n-variate multilinear polynomial $f(X_1, \ldots, X_n) \in \mathbb{F}[X_1, \ldots, X_n]$ is uniquely determined by its evaluations on the vertices of the n-dimensional Boolean hypercube, we can split f into $M = 2^m$ sub-polynomials $f^{(i)}(X_1, \ldots, X_{n-m}) = f(X_1, \ldots, X_{n-m}, \mathrm{bin}(i))$, where $\mathrm{bin}(i) \in \{0,1\}^m$ is the binary representation of $i \in [M]$, i.e., $f^{(i)}$ is the restriction of f with the last m variables fixed to the bits of $\mathrm{bin}(i)$. Then, we can represent f as follows:

$$f(X_1, \ldots, X_n) = \sum_{\mathbf{x} \in \{0,1\}^n} f(\mathbf{x}) \widetilde{eq}(\mathbf{x}, X_1, \ldots, X_n) = \sum_{i \in [M]} \sum_{\mathbf{x} \in \{0,1\}^{n-m}} f^{(i)}(\mathbf{x}) \widetilde{eq}(\mathbf{x} || \,\mathrm{bin}(i), X_1, \ldots, X_n),$$

where $\widetilde{eq}(\mathbf{x}, X_1, \ldots, X_n) = \prod_{j=1}^n (x_j X_j + (1-x_j)(1-X_j))$ is simply the Lagrange basis polynomial for multilinear interpolation over the Boolean hypercube corresponding to the evaluation point $\mathbf{x} \in \{0,1\}^n$.

The commitment key contains evaluations of the multilinear Lagrange bases w.r.t. the n-dimensional Boolean hypercube on a secret evaluation point $\mathbf{t} \leftarrow \mathbb{F}_p^n$, i.e., $([\widetilde{eq}(\mathbf{x}, \mathbf{t})]_1)_{\mathbf{x} \in \{0,1\}^n}$. In the commitment phase, each sub-prover P_i locally commits to its polynomial $f^{(i)}(X_1, \ldots, X_{n-m})$, and sends the commitment $\mathrm{com}_f^{(i)}$ to the master node P_0. The partial commitments are then aggregated by P_0 into a global commitment $\mathrm{com}_f = \sum_{i \in [M]} \mathrm{com}_f^{(i)}$, which is sent to the verifier.

To provide an evaluation proof for value $v = f(\mathbf{r})$ at an evaluation point $\mathbf{r}$, each sub-prover P_i computes all its quotient polynomials $Q_k^{(i)}(\mathbf{x})$ w.r.t. the evaluation point $\mathbf{r}$ for all $1 \leq k \leq n-m$, computes the commitments $\mathrm{com}_{Q_k}^{(i)}$ to all of its quotient polynomials, and sends them to the master prover. P_0 then

$\mathsf{deMKZG.KGen}\left(1^\lambda, 1^n, \mathsf{aux} = \mathsf{gp} = (p, [1]_1, [1]_2, \bullet)\right)$:

Sample $\mathbf{t} \leftarrow \mathbb{F}_P^n$.

Output $\mathsf{ck} = \left(\mathsf{gp}, (\widetilde{eq}(\mathbf{x}, \mathbf{t})]_1)_{\mathbf{x}\in\{0,1\}^n}, ([t_i]_2)_{i\in[1,n]}\right)$, where $\widetilde{eq}(\mathbf{x}, \mathbf{t}) = \prod_{j=1}^n (x_j t_j + (1 - x_j)(1 - t_j))$.

$\mathsf{deMKZG.Com}(\mathsf{ck}, f)$:

1. For all $i \in [M]$, compute $\mathrm{com}_f^{(i)} = \sum_{\mathbf{x}\in\{0,1\}^{n-m}} f^{(i)}(\mathbf{x})[\widetilde{eq}(\mathbf{x}||\,\mathrm{bin}(i), \mathbf{t})]_1$, where $f^{(i)}(X_1, \ldots, X_{n-m}) = f(X_1, \ldots, X_{n-m}, \mathrm{bin}(i))$.
2. Compute $\mathrm{com}_f = \sum_{i\in[M]} \mathrm{com}_f^{(i)}$.

Output com_f.

$\mathsf{deMKZG.Open}(\mathsf{ck}, f, \mathbf{r} = (r_1, \ldots, r_n))$:

1. For all $k \in [1, n-m]$ and all $i \in [1, M]$ compute:
   - $R_k^{(i)}(\mathbf{x}) = (1 - r_k) \cdot R_{k-1}^{(i)}(0, \mathbf{x}) + r_k \cdot R_{k-1}^{(i)}(1, \mathbf{x}),\ \forall \mathbf{x} \in \{0,1\}^{n-k}$,
   - $Q_k^{(i)}(\mathbf{x}) = R_{k-1}^{(i)}(1, \mathbf{x}) - R_{k-1}^{(i)}(0, \mathbf{x}),\ \forall \mathbf{x} \in \{0,1\}^{n-k}$,
   - the commitment for $Q_k^{(i)}$ as $\mathrm{com}_{Q_k}^{(i)}$.
2. For all $k = 1, \ldots, n-m$, compute $\mathrm{com}_{Q_k} = \sum_{i\in[M]} \mathrm{com}_{Q_k}^{(i)}$.
3. Construct the evaluation table for Q_{n-m} and R_{n-m}.
4. For all $k \in [n-m+1, n]$ compute:
   - $R_k(\mathbf{x}) = (1 - r_k) \cdot R_{k-1}(0, \mathbf{x}) + r_k \cdot R_{k-1}(1, \mathbf{x})$ for all $\mathbf{x} \in \{0,1\}^{n-k}$,
   - $Q_k(\mathbf{x}) = R_{k-1}(1, \mathbf{x}) - R_{k-1}(0, \mathbf{x}),\ \forall \mathbf{x} \in \{0,1\}^{n-k}$.
   - Compute the commitment for Q_k as com_{Q_k}.

Output $(f(\mathbf{r}), (\mathrm{com}_{Q_1}, \ldots, \mathrm{com}_{Q_n}))$.

$\mathsf{deMKZG.Ver}(\mathsf{ck}, \mathrm{com}_f, \mathbf{r} = (r_1, \ldots, r_n), v, \pi = (\mathrm{com}_{Q_1}, \ldots, \mathrm{com}_{Q_n}))$:

If $(\mathrm{com}_f - [v]_1) \bullet [1]_2 = \bigotimes_{i=1}^n \mathrm{com}_{Q_i} \bullet [t_i - r_i]$ output 1, otherwise 0.

Fig. 8. The deMKZG Polynomial Commitment Scheme.

combines these partial commitments from sub-provers into the final commitments $\mathrm{com}_{Q_k} = \sum_{i\in[M]} \mathrm{com}_{Q_k}^{(i)}$, for all $1 \le k \le n-m$. Then, for $n-m+1 \le k \le n$, the master prover computes the commitments to the last m quotient polynomials com_{Q_k} on its own and sends the complete proof $\pi = (\mathrm{com}_{Q_1}, \ldots, \mathrm{com}_{Q_n})$ along with the value $v = f(\mathbf{r})$ to the verifier. The verifier accepts the evaluation proof based on the check

$$(\mathrm{com}_f - [v]_1) \bullet [1]_2 = \bigotimes_{i=1}^n \mathrm{com}_{Q_i} \bullet [t_i - r_i],$$

which is exactly the multivariate KZG check.

While the computation of the quotient polynomials is performed differently, the underlying structure of the deMKZG scheme remains exactly the same as in the basic multilinear KZG scheme. Thus, the proof of extractability of deMKZG follows directly from Theorem 7.

Theorem 9. *Let* $\mathbf{d} = (1, \ldots, 1) \in \mathsf{poly}(\lambda)$ *be degree bounds specifying the set of* n*-variate multilinear polynomials and* PGen *be a bilinear group generator.*

1. *If the* ARSDH(n) *assumption holds for* PGen *then the extended* deMKZG PoKoP *is a proof of knowledge of a polynomial for* deMKZG *with respect to* PGen *and* $\mathbf{d}$.
2. *If the* GARSDH(n) *assumption holds for* PGen *then both extended and canonical* deMKZG PoKoP*s are proofs of knowledge of a polynomial for* deMKZG *with respect to* PGen *and* $\mathbf{d}$.

7.3 Randomized KZG from [PST13]

Rather than standard evaluation binding, [PST13] proved a slightly weaker notion of evaluation binding *on average* for the multivariate KZG scheme presented in Figure 2 (see Section 10 for further discussion). Additionally, they introduced a *randomized* multivariate KZG scheme based on the following *parameterized*

multivariate polynomial decomposition lemma, for which they proved the standard notion of evaluation binding under the Strong Diffie-Hellmann assumption in bilinear groups.

Lemma 8 (Parameterized polynomial decomposition [PST13]). *Let $f(X_1, \ldots, X_n) \in \mathbb{F}_p[X_1, \ldots, X_n]$ be an n-variate polynomial. For all $(x_1, \ldots, x_n) \in \mathbb{F}_p^n$ and $(\alpha_1, \ldots, \alpha_{n-1}) \in \mathbb{F}_p^{n-1}$ such that $\prod_{i=1}^{n-1} \alpha_i \neq 0$, it holds that*

$$f(X_1, \ldots, X_n) - f(x_1, \ldots, x_n) = \sum_{i=1}^{n-1} (\alpha_i(X_i - x_i) + X_{i+1} - x_{i+1}) q_i(X_i, \ldots, X_n) + q_n(X_n)(X_n - x_n), \quad (14)$$

where $q_i(X_i \ldots, X_n) \in \mathbb{F}_p[X_i, \ldots, X_n]$ for all $i \in [1, n]$.

In the randomized multivariate KZG scheme from [PST13] based on the lemma above, the procedures KGen and Com are the same as in KZG. The construction of an evaluation proof for value $z = f(\mathbf{x})$ at an evaluation point $\mathbf{x}$ in the randomized scheme can be seen as the following interactive protocol between the prover and the verifier: The verifier samples $(\alpha_1, \ldots, \alpha_{n-1}) \leftarrow \mathbb{F}_p^{n-1}$ uniformly and sends it to the prover.[5] The prover then uses polynomial division to compute the quotient polynomials $\{q_i(X_i \ldots, X_n)\}_{i=1}^n$ implicitly defined for all $1 \leq i \leq n-1$ as

$$r_i(X_{i+1}, \ldots, X_n) = (\alpha_i(X_{i+1} - x_{i+1}) + X_{i+2} - x_{i+2})\, q_{i+1}(X_{i+1}, \ldots, X_n) + r_{i+1}(X_{i+2}, \ldots, X_n),$$

where $r_0(X_1, \ldots, X_n) = f(X_1, \ldots, X_n) - f(\mathbf{x})$. Finally, the last quotient polynomial is defined via $r_{n-1}(X_n) = (X_n - x_n)q_n(X_n)$ (By Lemma 8, we have that $r_n = 0$). Analogously to the standard KZG scheme, the prover then computes the evaluation proof as a vector of commitments to the n quotient polynomials, i.e., $\pi = ([q_1(t_1, \ldots, t_n)]_1, \ldots, [q_n(t_n)]_1)$, and sends (z, π) to the verifier. The verification check corresponding to Equation (14) has the form:

$$[f(\mathbf{t}) - z]_1 \bullet [1]_2 \stackrel{?}{=} \bigotimes_{i=1}^{n-1} \pi_i \bullet [\alpha_i(t_i - x_i) + t_{i+1} - x_{i+1}]_2 \otimes \pi_n \bullet [t_n - x_n]_2. \quad (15)$$

The above check has a different structure than the verification check in KZG. Thus, it is a priori unclear whether our results on the extractability of the KZG PoKoPs imply directly anything for the randomized multivariate scheme from [PST13] described above.

However, we show that eq. (14) can be rearranged to support a verification check with the same structure as in KZG and use this observation to prove the extractability of the randomized scheme. By redistributing the summation and combining terms containing $(X_i - x_i)$, we can rewrite eq. (14) as

$$f(\mathbf{X}) - f(\mathbf{x}) = \alpha_1 q_1(\mathbf{X})(X_1 - x_1) + \left(\sum_{i=2}^{n-1} (q_{i-1}(\mathbf{X}) + \alpha_i q_i(\mathbf{X}))(X_i - x_i) \right) + (q_{n-1}(\mathbf{X}) + q_n(\mathbf{X}))(X_n - x_n). \quad (16)$$

We can define alternative quotient polynomials $\widetilde{q}_i(\mathbf{X})$ for $1 \leq i \leq n$ as follows:

- $\widetilde{q}_1(X_1, \ldots, X_n) = \alpha_1 q_1(X_1, \ldots, X_n)$,
- $\widetilde{q}_i(X_{i-1}, \ldots, X_n) = q_{i-1}(X_{i-1}, \ldots, X_n) + \alpha_i q_i(X_i, \ldots, X_n)$ for $i \in [2, n-1]$,
- $\widetilde{q}_n(X_{n-1}, X_n) = q_{n-1}(X_{n-1}, X_n) + q_n(X_n)$.

By eq. (16), for the evaluation proof $\widetilde{\pi} = ([\widetilde{q}_1(t_1, \ldots, t_n)]_1, \ldots, [\widetilde{q}_n(t_{n-1}, t_n)]_1)$ computed as the vector of commitments to the alternative quotient polynomials, the verification check can be performed as

$$[f(\mathbf{t}) - z]_1 \bullet [1]_2 \stackrel{?}{=} \bigotimes_{i=1}^{n} \widetilde{\pi}_i \bullet [t_i - x_i],$$

[5] To avoid cumbersome notation, we do not exclude 0 when sampling the α_i's. We can afford this since the probability of sampling any 0 is at most $(n-1)/|\mathbb{F}_p|$, which is exponentially small in the security parameter λ.

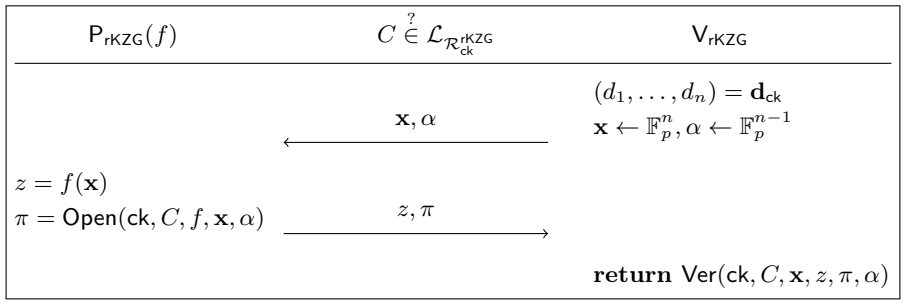

Fig. 9. The canonical PoKoP for $\mathsf{rKZG} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$.

which is exactly the original multivariate KZG check. Importantly, the verifier can efficiently compute $\widetilde{\pi}$ from π using the vector of challenges α since, by the definition of the polynomials $\widetilde{q}_i$, it holds that

$$\widetilde{\pi} = (\alpha_1\pi_1, \pi_1 + \alpha_2\pi_2, \ldots, \pi_{n-2} + \alpha_{n-1}\pi_{n-1}, \pi_{n-1} + \pi_n). \tag{17}$$

To summarize, for the randomized KZG scheme from [PST13], there is an equivalent scheme $\mathsf{rKZG} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$, where the procedures KGen and Com are the same as in KZG. The evaluation proof is constructed by $\mathsf{rKZG.Open}(\mathsf{ck}, C, f, \mathbf{x}, \alpha)$ as a KZG evaluation proof using the quotient polynomials q_i computed w.r.t. α via the parameterized polynomial decomposition lemma (Lemma 8). The verification check $\mathsf{rKZG.Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi, \alpha)$ first computes $\widetilde{\pi}$ defined in eq. (17) and then outputs $\mathsf{KZG.Ver}(\mathsf{ck}, C, \mathbf{x}, z, \widetilde{\pi})$. Next, we discuss the extractability of the rKZG scheme.

In Figure 9, we present the canonical PoKoP for rKZG. Note that the first round still slightly differs from the canonical KZG PoKoP. Specifically, verifier samples a uniform evaluation point $\mathbf{x} = (x_1, \ldots, x_n) \leftarrow \mathbb{F}_q^n$ and a randomization vector $\alpha = (\alpha_1, \ldots, \alpha_{n-1}) \leftarrow \mathbb{Z}_p^{n-1}$ and sends them to the prover. We can establish the knowledge-soundness of the canonical PoKoP for GARSDH, as stated in the following theorem.

Theorem 10. *Let* $\mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda)$ *and* $\mathsf{ck} \leftarrow \mathsf{rKZG.KGen}(1^\lambda, (d_1, \ldots, d_n), \mathsf{gp})$ *for some* $d_1, \ldots, d_n \in \mathbb{N}$. *If the* $\mathsf{GARSDH}(n)$ *assumption holds for* PGen *then the canonical* rKZG PoKoP *is an interactive argument of knowledge for* $\mathcal{R}_{\mathsf{ck}}^{\mathsf{rKZG}}$.

Proof. If we want to proceed along the lines of the proof of Theorem 7, we have to at first establish that if the $\mathsf{GARSDH}(n)$ assumption holds, then the canonical randomized KZG PoKoP is a restricted $(d_1+1)\ldots(d_n+1)$-special sound interactive argument for the relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{rKZG}}$ w.r.t. the class of tree constructing algorithms that output consistent trees. To do this, we construct a PPT extractor Ext. Note that the transcripts look different now, namely each transcript now contains also a randomization vector α

$$tr^{(\mathbf{i})} = \left(\mathsf{ck}, [c]_1, x_1^{(i_1)}, x_2^{(i_1,i_2)}, \ldots, x_n^{(i_1,\ldots,i_n)}, [\varphi_1^{(\mathbf{i})}]_1, \ldots, [\varphi_n^{(\mathbf{i})}]_1, z^{(\mathbf{i})}, \alpha\right).$$

However, we can see that this does not change the extractor from the one we have constructed in the Theorem 5, because for the interpolation, we do not need α. Therefore the extractor is the same as in Figure 6. We can continue identically as in Theorem 5 to get the same result.

By the above, we know that if the $\mathsf{GARSDH}(n)$ assumption holds, then the canonical randomized KZG PoKoP is a restricted $((d_1+1)\cdots(d_n+1))$-special sound interactive argument of knowledge for the relation $\mathcal{R}_{\mathsf{ck}}^{\mathsf{rKZG}}$. To prove knowledge-soundness of the canonical randomized KZG PoKoP it remains to construct a PPT algorithm that outputs $((d_1+1)\cdots(d_n+1))$ accepting transcripts that are consistent, given black-box access to an adversary $\mathcal{A}$ that on input (ck, C) convinces the verifier $\mathsf{V}_{\mathsf{KZG}}$ with probability δ.

Define $H \in \{0,1\}^{R \times p \times \ldots \times p \times (p-1)^n}$ as follows. Element $H_{i_0,i_1,\ldots,i_n,j}$ contains $\mathsf{rKZG.Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi, \alpha)$, where i_0 denotes the randomness of $\mathcal{A}$ (which determines z and π), $i_j = x_j$ for all $j \in [1, n]$ and j denotes the randomization vector α. We can emulate an oracle for H by interacting with $\mathcal{A}$. □

8 (G)ARSDH in AGM and GGM

In this section, we prove hardness of the ARSDH and GARSDH assumptions in two idealized models: Shoup's Generic Group Model (GGM) [Sho97] and the Algebraic Group Model (AGM) [FKL18]. In Shoup's GGM [Sho97], we assume that an adversary only sees uniformly random labels of group elements and gets access to an oracle that can be queried on two labels δ_1, δ_2 and outputs the label of the group element obtained by performing the group operation on the elements corresponding to δ_1 and δ_2. In contrast, in the AGM [FKL18] the adversary does have access to the group's representation. However, whenever the adversary outputs a group element, it must explain how it obtained this element from the input elements. Due to the strong restriction on the adversary, the AGM allows for simple reductions between hardness assumptions we do not know of in the standard model. The GGM enables us to compute lower bounds on the number of group operations needed to break an assumption. For a detailed comparison of the two models see, e.g., [Zha22].

8.1 Hardness of ARSDH and GARSDH in the AGM

In the AGM, we assume an (algebraic) adversary that has access to the group and can see the representation of the elements of the group. Since the adversary is not confined to work with just labels as is the case of the GGM, it can plausibly exploit the specific structure of the group. However, for every group element it outputs, the adversary must output its representation with respect to all of the previous group elements. Specifically, assume the adversary $\mathcal{A}$ received input elements $([x_1]_1, \ldots, [x_n]_1)$, then whenever $\mathcal{A}$ outputs an element $[x]_1 \in \mathbb{G}_1$, it also has to output elements $\alpha_i \in \mathbb{F}$, such that $[x]_1 = \alpha_1 \cdot [x_1]_1 + \ldots + \alpha_n \cdot [x_n]_1$ holds, i.e., the adversary always outputs $[x]_1$ together with the explanation vector $(\alpha_1, \ldots, \alpha_n)$. The fact that algebraic adversaries output a linear representation of each output element is very helpful in reductions between hardness assumptions.

The common approach of reductions in the AGM is to reduce the hardness of one assumption to the hardness of a variant of the Power Discrete Logarithm (PDL) assumption. This assumption states that, given $q+1$ group elements of the form $[1], [x], [x^2], \ldots, [x^q]$, it is hard to find x. Below, we define the PDL assumption in bilinear groups.

Definition 20 ((d_1, d_2)-PDL assumption). *Let* PGen *be a bilinear group generator, and* $d_1(\lambda), d_2(\lambda) \in \mathsf{poly}(\lambda)$. *We say the* (d_1, d_2)-Power Discrete Logarithm ((d_1, d_2)-PDL) assumption *holds for* PGen *if, for all* PPT $\mathcal{A}$,

$$\Pr\left[\mathcal{A}\left(\mathsf{gp}, \left([\sigma^i]_1\right)_{i=0}^{d_1}, \left([\sigma^i]_2\right)_{i=0}^{d_2}\right) = \sigma \;\middle|\; \mathsf{gp} \leftarrow \mathsf{PGen}(1^\lambda); \sigma \leftarrow \mathbb{F}_p^* \right] \leq \mathsf{negl}(\lambda).$$

In a typical security proof in the Algebraic Group Model (AGM), we analyze a polynomial $V(X)$, called the verification polynomial. This polynomial comes from the requirement that an algebraic adversary must provide a linear representation for any group element it outputs. Because of this constraint, we can express the adversary's responses as polynomial equations involving the discrete logarithms of the input group elements. The main idea of the proof is that if this polynomial equals zero at some point, then the assumption being analyzed has been broken. In the proof, we consider two cases:

1. Case 1: $V(X)$ is identically zero:
   If $V(X) = 0$ for all values of X, it means the adversary's responses always satisfy the verification equation, regardless of the input. We show that this case is impossible.
2. Case 2: $V(X) \neq 0$, but $V(x) = 0$ for some specific x:
   In this case, we assume that $V(X)$ is a nonzero polynomial, but there exists some value x such that $V(x) = 0$. This allows us to reduce the problem to a known computational hardness assumption. In the AGM section, we are using the $(n, 1)$-PDL (Power Discrete Logarithm) assumption.

Now, we show how a proof of security in AGM proceeds on an example from [LPS23]. We are asked to prove that solving the Computational Diffie-Hellman (CDH) (given $\mathsf{gp}, [a]_1, [b]_1$, it is intractable to find

$[ab]_1$) problem in AGM is not easier than solving the Discrete Logarithm (DL) problem (given $\mathsf{gp}, [a]_1$, it is intractable to find a).

The adversary $\mathcal{A}$ is given input elements from a group $\mathbb{G}_1$. Specifically, $\mathcal{A}$ receives: $[1]_1, [a]_1, [b]_1$, where $[1]_1$ is the generator of the group, $[a]_1$ is a group element whose discrete logarithm is a, and $[b]_1$ is another group element whose discrete logarithm is b. The adversary's goal is to compute the element $[ab]_1$, which is the result of multiplying the discrete logarithms of the given elements. Since $\mathcal{A}$ is algebraic, it cannot simply output a group element as its response. Instead, it must also provide integer coefficients v_1, v_2, v_3 such that $[ab]_1 = v_1[1]_1 + v_2[a]_1 + v_3[b]_1$ holds. This equation expresses $[ab]_1$ as a linear combination of the known group elements.

Now, let us construct a reduction algorithm $\mathcal{B}$ that solves the Discrete Logarithm problem. The reduction works as follows. $\mathcal{B}$ receives the DL challenge, which is to compute a given $[a]_1$ and $[1]_1$. It chooses a random integer b and sets up the CDH problem by computing $[b]_1$. It then runs the algebraic adversary $\mathcal{A}$, providing $[1]_1, [a]_1, [b]_1$ as input. If $\mathcal{A}$ successfully outputs $[ab]_1$, it must also provide the coefficients v_1, v_2, v_3, satisfying equation $[ab]_1 = v_1[1]_1 + v_2[a]_1 + v_3[b]_1$. This equation can be rewritten in terms of discrete logarithms $ab = v_1 + v_2 a + v_3 b$. Rearranging, we obtain the polynomial $V(X) = v_1 + v_2 X + v_3 b - Xb$. If $\mathcal{A}$ successfully solves the CDH problem, then this polynomial must have a as a root. The reduction $\mathcal{B}$ can now find a by finding the root of the polynomial $V(X)$.

The key insight is that the algebraic adversary is forced to express its output in a specific way, making it possible to extract useful information. If $\mathcal{A}$ can solve CDH, then the verification polynomial $V(X)$ must satisfy certain conditions, which allows $\mathcal{B}$ to extract the discrete logarithm. By leveraging the structure imposed by the AGM, we can show that solving the CDH problem cannot be easier than solving the DL problem in AGM.

First, we show that $\mathsf{ARSDH}(n)$ is equivalent to PDL in the AGM.

Lemma 9 (PDL reduces to $\mathsf{ARSDH}(n)$ in AGM). *Let PGen be a bilinear group generator. If the $(k,1)$-PDL assumption holds for PGen for all $k \in \mathsf{poly}(\lambda)$, then the $\mathsf{ARSDH}(n)$ holds in the AGM for PGen for all $n \in \mathsf{poly}(\lambda)$.*

Proof. Let gp be sampled by PGen and, for $k \in \mathsf{poly}(\lambda)$, let $\left(\mathsf{gp}, [1]_1, [\sigma]_1, \ldots, [\sigma^k]_1, [1]_2, [\sigma]_2\right)$ be a $(k,1)$-PDL instance. For arbitrary $n \in \mathsf{poly}(\lambda)$, we show how to reduce the above PDL instance to an $\mathsf{ARSDH}(n)$ instance with degree bounds $\mathbf{d} = (d_1, \ldots, d_n) = (k, \ldots, k)$. Recall that the input for an $\mathsf{ARSDH}(n)$ adversary is a tuple $\left(\mathsf{gp}, \left(\left[t_1^{i_1} \cdots t_n^{i_n}\right]_1\right)_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, [1]_2, ([t_i]_2)_{i\in[1,n]}\right)$ for $(t_1, \ldots, t_n) \leftarrow \mathbb{F}_p^n$. To embed the above PDL instance to an $\mathsf{ARSDH}(n)$ instance, we set $t_1 = \sigma$ and sample the remaining t_i uniformly at random from $\mathbb{F}_p$ for $2 \le i \le n$. Then, we compute the instance as $\left(\mathsf{gp}, \left(\left[\sigma^{i_1} t_2^{i_2} \cdots t_n^{i_n}\right]_1\right)_{i_1,\ldots,i_n=0}^{k,\ldots,k}, [1]_2, [\sigma]_2, ([t_i]_2)_{i\in[2,n]}\right)$ and give it as input to an algebraic adversary for $\mathsf{ARSDH}(n)$. Note that since we sampled $t_2, \ldots, t_n$, we can compute all the cross terms $\left[\sigma^{i_1} t_2^{i_2} \cdots t_n^{i_n}\right]_1$ simply as $(t_2^{i_2} \cdots t_n^{i_n})[\sigma^{i_1}]_1$. With a non-negligible probability, the algebraic $\mathsf{ARSDH}(n)$ adversary wins the $\mathsf{ARSDH}(n)$ game, i.e., it returns

$$[\chi]_1, \quad \{[\varphi_i]_1\}_{i=1}^{n-1}, \quad \{x_i\}_{i=1}^{n-1} \subseteq \mathbb{F}_p, \quad [\psi]_1, \quad \text{and} \quad S \subseteq \mathbb{F}_p,$$

and the explanations of all the output group elements w.r.t. its input, such that

- $|S| = k+1$,
- $[\chi]_T \neq \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2$, and
- $[\chi]_1 \bullet [1]_2 = \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2 \otimes [\psi]_1 \bullet [Z_S(t_n)]_2$.

We use the explanations the adversary provided to construct an n-variate verification polynomial

$$V(X_1, \ldots, X_n) = \chi(X_1, \ldots, X_n) - \sum_{i=1}^{n-1} \varphi_i(X_1, \ldots, X_n) \cdot (X_i - x_i) - \psi(X_1, \ldots, X_n) \cdot Z_S(X_n).$$

The verification check $[\chi]_1 \bullet [1]_2 = \bigotimes_{i=1}^{n-1} [\varphi_i]_1 \bullet [t_i - x_i]_2 \otimes [\psi]_1 \bullet [Z_S(t_n)]_2$ is equivalent to $V(t_1, \ldots, t_n) = 0$, which can happen in two ways:

$V(X_1,\ldots,X_n) \equiv 0$ **is the identically zero polynomial:** In this case, we get the polynomial identity

$$\frac{\chi(X_1,\ldots,X_n) - \sum_{i=1}^{n-1} \varphi_i(X_1,\ldots,X_n)\cdot(X_i - \alpha_i)}{Z_S(X_n)} = \psi(X_1,\ldots,X_n).$$

Thus, $Z_S(X_n)$ must divide $\chi(X_1,\ldots,X_n) - \sum_{i=1}^{n-1} \varphi_i(X_1,\ldots,X_n)\cdot(X_i - x_i)$ without remainder since $\psi(X_1,\ldots,X_n)$ is a polynomial. Note that the degree of X_n in $\chi(X_1,\ldots,X_n) - \sum_{i=1}^{n-1} \varphi_i(X_1,\ldots,X_n)\cdot(X_i - x_i)$ is at most k while the degree of X_n in $Z_S(X_n)$ is $k+1$, and thus $\chi(X_1,\ldots,X_n) - \sum_{i=1}^{n-1} \varphi_i(X_1,\ldots,X_n)\cdot(X_i - x_i)$ must be the identically zero polynomial. However, this is in contradiction with the second condition in the $\mathsf{ARSDH}(n)$ game, which states that $[\chi]_T \neq \bigotimes_{i=1}^{n-1}[\varphi_i]_1 \bullet [t_i - x_i]_2$. Hence, this case cannot occur.

$V(X_1,\ldots,X_n) \not\equiv 0$**, but** $V(t_1,\ldots,t_n) = 0$**:** Since we have set $t_1 = \sigma$, we have that $V(\sigma, t_2,\ldots,t_n) = 0$. Consider the univariate polynomial $V^*(X_1) = V(X_1, t_2,\ldots,t_n)$, which depends only on the first variable. Since $V^*(\sigma) = 0$, we can compute σ and break the $(k,1)$-PDL assumption by factoring V^*. □

Next, we show that $\mathsf{GARSDH}(n)$ is also equivalent to PDL in the AGM.

Lemma 10 (PDL **reduces to** $\mathsf{GARSDH}(n)$ **in AGM).** *Let* PGen *be a bilinear group generator. If the* $(k,1)$*-PDL assumption holds for* PGen *for all* $k \in \mathsf{poly}(\lambda)$*, then the* $\mathsf{GARSDH}(n)$ *holds in the AGM for* PGen *for all* $n \in \mathsf{poly}(\lambda)$*.*

Proof. The proof is similar to the above proof of Lemma 9. The main difference is that, due to the different winning condition in GARSDH, we get a slightly different n-variate verification polynomial $V(X_1,\ldots,X_n)$. Let gp be sampled by PGen and, for $k \in \mathsf{poly}(\lambda)$, let $\left(\mathsf{gp}, [1]_1, [\sigma]_1, \ldots, [\sigma^k]_1, [1]_2, [\sigma]_2\right)$ be a $(k,1)$-PDL instance. For arbitrary $n \in \mathsf{poly}(\lambda)$, we show how to reduce the above PDL instance to a $\mathsf{GARSDH}(n)$ instance with degree bounds $\mathbf{d} = (d_1,\ldots,d_n) = (k,\ldots,k)$. Recall that the input for an $\mathsf{GARSDH}(n)$ adversary is a tuple $\left(\mathsf{gp}, \left(\left[t_1^{i_1}\cdots t_n^{i_n}\right]_1\right)_{i_1,\ldots,i_n=0}^{d_1,\ldots,d_n}, [1]_2, ([t_i]_2)_{i\in[1,n]}\right)$ for $(t_1,\ldots,t_n) \leftarrow \mathbb{F}_p^n$. To embed the above PDL instance to an $\mathsf{ARSDH}(n)$ instance, we set $t_1 = \sigma$ and sample the remaining t_i uniformly at random from $\mathbb{F}_p$ for $2 \le i \le n$. Then, we compute the instance as $\left(\mathsf{gp}, \left(\left[\sigma^{i_1} t_2^{i_2}\cdots t_n^{i_n}\right]_1\right)_{i_1,\ldots,i_n=0}^{k,\ldots,k}, [1]_2, [\sigma]_2, ([t_i]_2)_{i\in[2,n]}\right)$ and give it as input to an algebraic adversary for $\mathsf{GARSDH}(n)$. Note that since we sampled $t_2,\ldots,t_n$, we can compute all the cross terms $\left[\sigma^{i_1} t_2^{i_2}\cdots t_n^{i_n}\right]_1$ simply as $(t_2^{i_2}\cdots t_n^{i_n})[\sigma^{i_1}]_1$. With a non-negligible probability, the algebraic $\mathsf{GARSDH}(n)$ adversary wins the $\mathsf{GARSDH}(n)$ game, i.e., it returns

$$[\chi]_1, \left\{\left\{S_j^{(i_1,\ldots,i_{j-1})}\right\}_{i_1\in[d_1],\ldots,i_{j-1}\in[d_{j-1}]}, \left\{[\varphi_j^{(i_1,\ldots,i_{j-1})}]_1\right\}_{i_1\in[d_1],\ldots,i_{j-1}\in[d_{j-1}]}\right\}_{j\in[1,n]}$$

and the explanations of all the output group elements w.r.t. its input, such that

- $|S_j^{(i_1,\ldots,i_{j-1})}| = k+1$ for all $j \in [1,n]$ and all $i_1 \in [k],\ldots,i_{j-1} \in [k]$,
- $[\chi]_T \neq \bigotimes_{j=2}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$, and
- $[\chi]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$.

We can use the explanations the adversary provided to construct an n-variate verification polynomial

$$V(X_1,\ldots,X_n) = \chi(X_1,\ldots,X_n) - \sum_{j=1}^{n} \sum_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \varphi_j^{(i_1,\ldots,i_{j-1})}(X_1,\ldots,X_n) Z_{S_j^{(i_1,\ldots,i_{j-1})}}(X_j).$$

The verification check $[\chi]_1 \bullet [1]_2 = \bigotimes_{j=1}^{n} \bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} \left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$ is equivalent to $V(t_1,\ldots,t_n) = 0$, which can happen in two ways:

$V(X_1, \ldots, X_n) \equiv 0$ **is the identically zero polynomial:** In this case, we get the polynomial identity

$$\frac{\chi(X_1, \ldots, X_n) - \sum_{j=2}^{n} \sum_{i_1, \ldots, i_{j-1}=0}^{d_1, \ldots, d_{j-1}} \varphi_j^{(i_1, \ldots, i_{j-1})}(X_1, \ldots, X_n) Z_{S_j^{(i_1, \ldots, i_{j-1})}}(X_j)}{Z_S(X_1)} = \varphi(X_1, \ldots, X_n).$$

Thus, $Z_S(X_1)$ must divide $\chi(X_1, \ldots, X_n) - \sum_{j=2}^{n} \sum_{i_1, \ldots, i_{j-1}=0}^{d_1, \ldots, d_{j-1}} \varphi_j^{(i_1, \ldots, i_{j-1})}(X_1, \ldots, X_n) Z_{S_j^{(i_1, \ldots, i_{j-1})}}(X_j)$ without remainder since $\varphi(X_1, \ldots, X_n)$ is a polynomial. Note that the degree of X_1 in $\chi(X_1, \ldots, X_n) - \sum_{j=2}^{n} \sum_{i_1, \ldots, i_{j-1}=0}^{d_1, \ldots, d_{j-1}} \varphi_j^{(i_1, \ldots, i_{j-1})}(X_1, \ldots, X_n) Z_{S_j^{(i_1, \ldots, i_{j-1})}}(X_j)$ is at most k while the degree of X_1 in $Z_S(X_1)$ is $k+1$, and thus $\chi(X_1, \ldots, X_n) - \sum_{j=2}^{n} \sum_{i_1, \ldots, i_{j-1}=0}^{d_1, \ldots, d_{j-1}} \varphi_j^{(i_1, \ldots, i_{j-1})}(X_1, \ldots, X_n) Z_{S_j^{(i_1, \ldots, i_{j-1})}}(X_j)$ must be the identically zero polynomial. However, this is in contradiction with the second condition in the $\mathsf{GARSDH}(n)$ game, which states that $[\chi]_T \neq \bigotimes_{j=2}^{n} \bigotimes_{i_1, \ldots, i_{j-1}=0}^{d_1, \ldots, d_{j-1}} \left[\varphi_j^{(i_1, \ldots, i_{j-1})}\right]_1 \bullet \left[Z_{S_j^{(i_1, \ldots, i_{j-1})}}(t_j)\right]_2$. Hence, this case cannot occur.

$V(X_1, \ldots, X_n) \not\equiv 0$**, but** $V(t_1, \ldots, t_n) = 0$**:** Since we have set $t_1 = \sigma$, we have that $V(\sigma, t_2, \ldots, t_n) = 0$. Consider the univariate polynomial $V^*(X_1) = V(X_1, t_2, \ldots, t_n)$, which depends only on the first variable. Since $V^*(\sigma) = 0$, we can compute σ and break the $(k,1)$-PDL assumption by factoring V^*. □

Lipmaa, Parisella, and Siim [LPS23] recently proposed the AGM with Oblivious Sampling (AGMOS), which is a strengthening of the algebraic group model that accounts for adversaries' possibility of sampling group elements without knowing their discrete logarithms, and, thus, not necessarily being able to explain the sampled group elements algebraically relative to group elements received as input. We believe our analyses of ARSDH and GARSDH in AGM would easily extend to AGMOS under the TOFR assumption suggested by [LPS23].

8.2 Hardness of ARSDH and GARSDH in the GGM

In the GGM, the adversary sees only labels of group elements and has access to an oracle that computes the group operations and outputs the label of the result. These labels do not contain any information about the group's structure since they are uniformly random. We can think of an adversary as a generic algorithm that operates as follows. At the beginning of the execution, it obtains some labels of group elements. During its execution, the adversary specifies labels, for example, δ_i corresponding to the group element g_i and δ_j corresponding to the group element g_j. The oracle then finds g_i and g_j if they exist, computes $g_i + g_j = g$ and outputs a label δ corresponding to the element g.

In the setting of bilinear groups the adversary has access to five oracles. The first three oracles compute group operations in groups $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ respectively. Then, it can query the oracle that computes the pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \mapsto \mathbb{G}_T$. Finally, we give the adversary access to an oracle that computes the isomorphism ψ from elements of $\mathbb{G}_2$ to elements of $\mathbb{G}_1$. Let us denote the order of the groups $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ by q. We denote the labeling functions of the three groups by ξ_1, ξ_2 and ξ_T respectively, where ξ_i for $i \in \{1, 2, T\}$ maps element in $\mathbb{G}_i$ to some (large enough) subset $E \subset \{0,1\}^k$, where $k > \lceil \log_2 q \rceil$. These mappings ξ_i can be thought of as mappings from $\mathbb{F}_p$ to the subset E of $\{0,1\}^k$, because we can identify each group element $[x]_i$ from $\mathbb{G}_i$ with its exponent x from $\mathbb{F}_p$.

In the rest of the section, we prove lower bounds on the query complexity of any adversary solving $\mathsf{ARSDH}(n)$ or $\mathsf{GARSDH}(n)$ in the GGM. Our proofs are similar to other lower bounds in GGM, such as [BB04].

Theorem 11 (Hardness of $\mathsf{ARSDH}(n)$ **in GGM).**
Let $\mathcal{A}$ *be an* $\mathsf{ARSDH}(n)$ *adversary for* $\boldsymbol{d} = (d_1, \ldots, d_n)$ *in the generic group model, making a total of at most* q *queries to the oracles that compute the group operations in* $\mathbb{G}_1$*,* $\mathbb{G}_2$*,* $\mathbb{G}_T$*, the oracle computing the isomorphism* ψ*, and the oracle computing the bilinear pairing* e*. If* $t_1, \ldots, t_n \in \mathbb{F}_p^n$ *and* ξ_1, ξ_2, ξ_T *are chosen uniformly at random, then the probability* ϵ *that*

$$\mathcal{A}(p, \xi_1(t_1^{i_1} \ldots t_n^{i_n})_{i_1 \in [d_1], \ldots, i_n \in [d_n]}, \xi_2(1), \xi_2(t_i)_{i \in [1,n]})$$

outputs $\left(\xi_1(\chi), \xi_1(\varphi_i)_{i\in[1,n-1]}, \xi_1(\psi), \{x_i\}_{i\in[1,n-1]}, S\right)$ *such that* $x_i \in \mathbb{F}_p$ *for all* $i \in [1, n-1]$, $S \subseteq \mathbb{F}_p$, $|S| = d_n + 1$, $[\chi]_T = \bigotimes_{i=1}^{n-1}[\varphi_i]_1 \bullet [t_i - x_i] \otimes [\psi]_1 \bullet [Z_S(t_n)]_2$, *and* $[\chi]_T \neq \bigotimes_{i=1}^{n-1}[\varphi_i]_1 \bullet [t_i - x_i]$ *is bounded by:*

$$\epsilon \leq \left(q + \prod_{i=1}^{n}(d_i + 1) + n + 1\right)^2 \frac{\sum_{i=1}^{n} d_i}{p^n} + \frac{\sum_{i=1}^{n} d_i + d_n}{p^n}.$$

Proof. Consider an algorithm $\mathcal{B}$ that plays the following game with $\mathcal{A}$. $\mathcal{B}$ keeps track of the following lists: $L_1 = \{(F_{1,i}, \xi_{1,i}) : i = 0, \ldots, \tau_1 - 1\}$, $L_2 = \{(F_{2,i}, \xi_{2,i}) : i = 0, \ldots, \tau_2 - 1\}$ and $L_T = \{(F_{T,i}, \xi_{T,i}) : i = 0, \ldots, \tau_T - 1\}$. Each list contains tuples $(F, \xi(F))$, of polynomials $F(X_1, \ldots, X_n) \in \mathbb{F}[X_1, \ldots, X_n]$ and binary strings $\xi(F) \in E \subset \{0,1\}^*$, where $\xi(F)$ is the encoding of the polynomial F. The generic adversary can only see the binary strings $\xi(F)$. At the beginning of the game, the algorithm $\mathcal{B}$ initializes all of the lists L_1, L_2 and L_T. To initialize the list L_1, $\mathcal{B}$ assigns every monomial $X_1^{i_1} \ldots X_n^{i_n}$ for $i_1 \in [d_1], \ldots, i_n \in [d_n]$ to a distinct polynomial $F_{1,i}$. The binary encodings of these polynomials $F_{1,i}$ are selected one by one, so that different polynomials correspond to different binary encodings. After $\mathcal{B}$ finishes initialization of the list L_1, there are $\tau_1 = (d_1 + 1) \cdots (d_n + 1)$ tuples. For the list L_2, the algorithm $\mathcal{B}$ initializes $F_{2,0}$ to 1, $F_{2,i}$ to X_i for all $i \in [1, n]$. $\mathcal{B}$ then computes the encodings of these polynomials using encoding ξ_2, so that binary strings for different polynomials are distinct. We can see that the size of the list L_2 after $\mathcal{B}$ finishes the initialization is $\tau_2 = n + 1$. The list L_T is initialized as empty, because $\mathcal{A}$ does not get any encodings of the elements of the target group. Therefore $\tau_T = 0$. At the beginning of the game, the sizes of the lists are: $\tau_1 = (d_1 + 1) \ldots (d_n + 1)$, $\tau_2 = n + 1$, and finally $\tau_T = 0$. During the game, the generic adversary will be querying five oracles. These oracles will be producing new polynomials and new encodings, so the lists L_1, L_2 and L_T will be augmented by $\mathcal{B}$. Below, we specify how these new polynomials are constructed. The polynomials $F_{1,i}$, $F_{2,i}$ are of degree $\leq d_i$ with respect to X_i for all $i \in \{1, \ldots, n\}$ and of total degree $\leq d_1 + \ldots + d_n$. The polynomials $F_{T,i}$ are of total degree $\leq 2(d_1 + \ldots + d_n)$. Note that for any $\xi_{1,i}, \xi_{2,i}$ and $\xi_{T,i}$, $\mathcal{B}$ can determine the $F_{1,i}, F_{2,i}$ and $F_{T,i}$, such that $\xi_{1,i} = \xi_1(F_{1,i}), \xi_{2,i} = \xi_1(F_{2,i})$ and $\xi_{T,i} = \xi_T(F_{T,i})$. After $\mathcal{B}$ provides $\mathcal{A}$ with the $\tau_1 + \tau_2 + \tau_T = (d_1 + 1) \cdots (d_n + 1) + n + 1$ binary strings, $\mathcal{A}$ starts sending queries to $\mathcal{B}$. These queries are either group operation queries, isomorphism queries, or pairing queries.

Group operation: Given a add/subtract selection bit and two operands $\xi_{1,i}, \xi_{1,j}$ compute:

$$F_{1,\tau_1} = F_{1,i} \pm F_{1,j} \in \mathbb{F}_p[X_1, \ldots, X_n].$$

If $F_{1,\tau_1} = F_{1,l}$ for some l that has been already defined, set $\xi_{1,\tau_1} = \xi_{1,l}$. Otherwise, assign ξ_{1,τ_1} a new random string from $E \setminus \{\xi_{1,0}, \ldots, \xi_{1,\tau_1-1}\}$. Add $(F_{1,\tau_1}, \xi_{1,\tau_1})$ to L_1, give ξ_{1,τ_1} to $\mathcal{A}$, and increment τ_1. Similar operations are performed for L_2 and L_T.

Isomorphism: Given a string $\xi_{1,i}$ with $0 \leq i < \tau_1$, we let:

$$F_{2,\tau_2} = F_{1,i} \in \mathbb{F}_p[X_1, \ldots, X_n].$$

If $F_{2,\tau_2} = F_{2,l}$ for some l already defined, we set $\xi_{2,\tau_1} = \xi_{2,l}$. Otherwise, we set ξ_{2,τ_2} to a string in $E \setminus \{\xi_{2,0}, \ldots, \xi_{2,\tau_2-1}\}$. We add $(F_{2,\tau_2}, \xi_{2,\tau_2})$ to L_2, give ξ_{2,τ_2} to $\mathcal{A}$, and increment τ_2 by one.

Pairing: Given $\xi_{1,i}$ and $\xi_{2,j}$ with i, j already defined, compute:

$$F_{T,\tau_T} = F_{1,i} \cdot F_{2,j} \in \mathbb{F}_p[X_1, \ldots, X_n].$$

If $F_{T,\tau_T} = F_{T,l}$ for some l that has been already defined, set $\xi_{T,\tau_T} = \xi_{T,l}$. Otherwise, assign a new random string, distinct from all of the previous strings. Add $(F_{T,\tau_T}, \xi_{T,\tau_T})$ to L_T, give ξ_{T,τ_T} to $\mathcal{A}$, and increment τ_T.

From the way how the algorithm $\mathcal{B}$ augments the lists, it is obvious that at step τ in the game, $\tau_1 + \tau_2 + \tau_T \leq \tau + (d_1 + 1) \ldots (d_n + 1) + n + 1$, because $\tau + (d_1 + 1) \ldots (d_n + 1) + n + 1$ is the maximum possible number of tuples in lists L_1, L_2 and L_T.

After at most q queries to oracles, $\mathcal{A}$ terminates and outputs the following tuple:

$$\left(\xi_1(\chi), \xi_1(\varphi_i)_{i\in[1,n-1]}, \xi_1(\psi), \{x_i\}_{i\in[1,n-1]}, S\right)$$

such that $|S| = d_n + 1$. There are some $F_{1,l}$, F_{1,l_i} for $i \in [1, n-1]$ and $F_{1,k}$ that correspond to $\xi_1(\chi)$, $\{\xi_1(\varphi_i)\}_{i\in[1,n-1]}$ and $\xi_1(\psi)$ respectively. We define the polynomial $F_{T,*}(X_1, \ldots, X_n)$ as

$$F_{T,*}(X_1, \ldots, X_n) = F_{1,l}(X_1, \ldots, X_n) - \sum_{i=1}^{n-1} F_{1,l_i}(X_1, \ldots, X_n)(X_i - x_i) - F_{1,k}(X_1, \ldots, X_n)Z_S(X_n).$$

At this stage, the algorithm $\mathcal{B}$ samples random $(t_1, \ldots, t_n) \in \mathbb{F}_p^n$. If $\mathcal{A}$'s answer is correct, then:

$$F_{T,*}(t_1, \ldots, t_n) = 0, \quad F_{1,l}(t_1, \ldots, t_n) \neq \sum_{i=1}^{n-1} F_{1,l_i}(t_1, \ldots, t_n)(t_i - x_i) \tag{3}$$

The Equation (3) corresponds to the adversary $\mathcal{A}$ breaking the $\mathsf{ARSDH}(n)$ instance (for $\mathbf{d} = (d_1, \ldots, d_n)$) in the discrete logarithms of the group elements.

There are two possibilities for $\mathcal{A}$ to solve the $\mathsf{ARSDH}(n)$ instance (for $\mathbf{d} = (d_1 \ldots, d_n)$).

Either $F_{T,*}(X_1, \ldots, X_n) = 0$ identically, or $F_{T,*}(X_1, \ldots, X_n) \neq 0$ and $(t_1, \ldots, t_n)$ is among the zeros of the polynomial $F_{T,*}(X_1, \ldots, X_n)$.

The equation $F_{T,*}(X_1, \ldots, X_n) = 0$ could be satisfied in two ways. Either the total degree of $F_{T,*}(X_1, \ldots, X_n)$ would be more than p^n, or $F_{T,*}(X_1, \ldots, X_n)$ would be equal to 0 identically. The first case could happen only if $d_1 + \ldots + d_{n-1} + 2d_n + 1 \geq p^n$, where $d_1 + \ldots + d_{n-1} + 2d_n + 1$ is the maximum possible total degree of $F_{T,*}(X_1, \ldots, X_n)$. The second case could happen, but only if $F_{1,k}(X_1, \ldots, X_n)$ is the identically zero polynomial. In this case, we would have

$$F_{1,l}(X_1, \ldots, X_n) - \sum_{i=1}^{n-1} F_{1,l_i}(X_1, \ldots, X_n)(X_i - x_i) = 0.$$

However this is in contradiction with the winning condition of the $\mathsf{ARSDH}(n)$ game. Hence, we can assume that there exists a tuple $(x'_1, \ldots, x'_n)$, such that $F_{T,*}(x'_1, \ldots, x'_n) \neq 0$.

Now the adversary $\mathcal{A}$ may win in two possible ways. Either the simulation provided by $\mathcal{B}$ is not perfect, in which case we can assume that $\mathcal{A}$ won, or substituting $(t_1, \ldots, t_n)$ into $(X_1, \ldots X_n)$ satisfies equation (3). The simulation provided by $\mathcal{B}$ is perfect unless the substitution of $(t_1, \ldots, t_n)$ into $(X_1, \ldots X_n)$ creates an equality relation between the simulated group elements that was not revealed to $\mathcal{A}$, during the query phase. Hence, the success probability of $\mathcal{A}$ is bounded by the probability of one of the following four events happening. The first three events correspond to the simulation provided by $\mathcal{B}$ to $\mathcal{A}$ not being perfect and the last event corresponds to $\mathcal{A}$ succeeding in breaking the $\mathsf{ARSDH}(n)$ instance for $\mathbf{d} = (d_1 \ldots, d_n)$.

1. $F_{1,i}(t_1, \ldots, t_n) = F_{1,j}(t_1, \ldots, t_n)$ for a pair of polynomials such that $i \neq j$.
2. $F_{2,i}(t_1, \ldots, t_n) = F_{2,j}(t_1, \ldots, t_n)$ for any a pair of polynomials such that $i \neq j$.
3. $F_{T,i}(t_1, \ldots, t_n) = F_{T,j}(t_1, \ldots, t_n)$ for a pair of polynomials such that $i \neq j$.
4. $F_{T,*}(t_1, \ldots, t_n) = 0$ and $F_{1,l}(t_1, \ldots, t_n) - \sum_{i=1}^{n-1} F_{1,l_i}(t_1, \ldots, t_n)(t_i - x_i) \neq 0$

The probabilities of the first three events are at most

$$\binom{\tau_1}{2}\frac{\sum_{i=1}^n d_i}{p^n}, \binom{\tau_2}{2}\frac{\sum_{i=1}^n d_i}{p^n}, \binom{\tau_T}{2}\frac{2(\sum_{i=1}^n d_i)}{p^n}$$

respectively. This follows from applying the Schwartz-Zippel lemma on the maximum possible degrees of the polynomials $F_{1,i}(X_1, \ldots, X_n), F_{2,i}(X_1, \ldots, X_n)$ and $F_{T,i}(X_1, \ldots, X_n)$. These maximum possible degrees are $\sum_{i=1}^n d_i, \sum_{i=1}^n d_i$ and $2(\sum_{i=1}^n d_i)$ respectively. For the analysis of the probability of event four, we can omit the condition that $F_{1,l}(t_1, \ldots, t_n) - \sum_{i=1}^{n-1} F_{1,l_i}(t_1, \ldots, t_n)(t_i - x_i) \neq 0$, since considering it, only lowers the overall probability of event four happening. Hence, the probability of event four is at most $(d_1 + \ldots + d_{n-1} + 2d_n + 1)/p^n$, where $(d_1 + \ldots + d_{n-1} + 2d_n + 1)$ is the maximum possible degree of the polynomial

$F_{T,*}(X_1,\ldots,X_n)$. We have obtained this upper bound by applying Schwartz-Zippel lemma on the maximum possible total degree of the polynomial $F_{T,*}(X_1,\ldots,X_n)$. $\mathcal{A}$ wins the game with probability at most

$$\begin{aligned}\epsilon &\leq \binom{\tau_1}{2}\frac{\sum_{i=1}^n d_i}{p^n} + \binom{\tau_2}{2}\frac{\sum_{i=1}^n d_i}{p^n} + \binom{\tau_T}{2}\frac{2(\sum_{i=1}^n d_i)}{p^n} + \frac{d_1+\ldots+d_{n-1}+2d_n+1}{p^n}\\ &\leq \tau_1^2\frac{\sum_{i=1}^n d_i}{p^n} + \tau_2^2\frac{\sum_{i=1}^n d_i}{p^n} + \tau_T^2\frac{2(\sum_{i=1}^n d_i)}{p^n} + \frac{d_1+\ldots+d_{n-1}+2d_n+1}{p^n}\\ &\leq (\tau_1+\tau_2+\tau_T)^2\frac{\sum_{i=1}^n d_i}{p^n} + \frac{\sum_{i=1}^n d_i + d_n}{p^n}.\end{aligned}$$

Using the fact that $\tau_1+\tau_2+\tau_T \leq q+(d_1+1)\ldots(d_n+1)+n+1$, we obtain the following upper bound on the probability:

$$\epsilon \leq \left(q+\prod_{i=1}^n (d_i+1)+n+1\right)^2\frac{\sum_{i=1}^n d_i}{p^n} + \frac{\sum_{i=1}^n d_i + d_n}{p^n},$$

which concludes the proof. □

Theorem 12 (Hardness of $\mathsf{GARSDH}(n)$ in GGM).
Let $\mathcal{A}$ be a $\mathsf{GARSDH}(n)$ adversary for $\boldsymbol{d}=(d_1,\ldots,d_n)$ in the generic group model, making a total of at most q queries to the oracles that compute the group operations in $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, the oracle computing the isomorphism ψ, and the oracle computing the bilinear pairing e. If $t_1,\ldots,t_n \in \mathbb{F}_p^n$ and ξ_1,ξ_2,ξ_T are chosen uniformly at random, then the probability ϵ that

$$\mathcal{A}(p,\xi_1(t_1^{i_1}\ldots t_n^{i_n})_{i_1\in[d_1],\ldots,i_n\in[d_n]},\xi_2(1),\xi_2(t_i)_{i\in[1,n]})$$

outputs a tuple

$$\left(\xi_1(\chi),\left\{\left\{S_j^{(i_1,\ldots,i_{j-1})}\right\}_{i_1\in[d_1],\ldots,i_{j-1}\in[d_{j-1}]},\left\{\xi_1\left(\varphi_j^{(i_1,\ldots,i_{j-1})}\right)\right\}_{i_1\in[d_1],\ldots,i_{j-1}\in[d_{j-1}]}\right\}_{j\in[1,n]}\right),$$

such that for all $j\in[1,n]$ and for all $i_1\in[d_1],\ldots,i_{j-1}\in[d_{j-1}]$, $S_j^{(i_1,\ldots,i_{j-1})}\subseteq\mathbb{F}_p^j$, $|S_j^{(i_1,\ldots,i_{j-1})}|=d_j+1$ and

$$[\chi]_1\bullet[1]_2 = \bigotimes_{j=1}^n\bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}}\left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1\bullet\left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$$

such that

$$[\chi]_T \neq \bigotimes_{j=2}^n\bigotimes_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}}\left[\varphi_j^{(i_1,\ldots,i_{j-1})}\right]_1\bullet\left[Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j)\right]_2$$

is bounded by:

$$\epsilon \leq \left(q+\prod_{i=1}^n (d_i+1)+n+1\right)^2\frac{\sum_{i=1}^n d_i}{p^n} + \frac{\sum_{i=1}^n d_i + d'}{p^n},$$

where $d'=\max\{d_1,\ldots,d_n\}$.

Proof. Consider an algorithm $\mathcal{B}$ that plays the following game with $\mathcal{A}$. $\mathcal{B}$ keeps track of the following lists: $L_1=\{(F_{1,i},\xi_{1,i}) : i=0,\ldots,\tau_1-1\}$, $L_2=\{(F_{2,i},\xi_{2,i}) : i=0,\ldots,\tau_2-1\}$ and $L_T=\{(F_{T,i},\xi_{T,i}) : i=0,\ldots,\tau_T-1\}$. Each list contains tuples $(F,\xi(F))$, of polynomials $F(X_1,\ldots,X_n)\in\mathbb{F}[X_1,\ldots,X_n]$ and binary strings $\xi(F)\in E\subset\{0,1\}^*$, where $\xi(F)$ is the encoding of the polynomial F. The generic adversary can only see the binary strings $\xi(F)$. At the beginning of the game, the algorithm $\mathcal{B}$ initializes all of the lists L_1,L_2 and L_T. To initialize the list L_1, $\mathcal{B}$ assigns every monomial $X_1^{i_1}\ldots X_n^{i_n}$ for $i_1\in[d_1],\ldots,i_n\in[d_n]$

to a distinct polynomial $F_{1,i}$. The binary encodings of these polynomials $F_{1,i}$ are selected one by one, so that different polynomials correspond to different binary encodings. After $\mathcal{B}$ finishes initialization of the list L_1, there are $\tau_1 = (d_1+1)\cdots(d_n+1)$ tuples. For the list L_2, the algorithm $\mathcal{B}$ initializes $F_{2,0}$ to 1, $F_{2,i}$ to X_i for all $i \in [1,n]$. $\mathcal{B}$ then computes the encodings of these polynomials using encoding ξ_2, so that binary strings for different polynomials are distinct. We can see that the size of the list L_2 after $\mathcal{B}$ finishes the initialization is $\tau_2 = n+1$. The list L_T is initialized as empty, because $\mathcal{A}$ does not get any encodings of the elements of the target group. Therefore $\tau_T = 0$. At the beginning of the game, the sizes of the lists are: $\tau_1 = (d_1+1)\ldots(d_n+1)$, $\tau_2 = n+1$, and finally $\tau_T = 0$. During the game, the generic adversary will be querying five oracles. These oracles will be producing new polynomials and new encodings, so the lists L_1, L_2 and L_T will be augmented by $\mathcal{B}$. Below, we specify how these new polynomials are constructed. The polynomials $F_{1,i}$, $F_{2,i}$ are of degree $\leq d_i$ with respect to X_i for all $i \in \{1,\ldots,n\}$ and of total degree $\leq d_1 + \ldots + d_n$. The polynomials $F_{T,i}$ are of total degree $\leq 2(d_1+\ldots+d_n)$. Note that for any $\xi_{1,i}, \xi_{2,i}$ and $\xi_{T,i}$, $\mathcal{B}$ can determine the $F_{1,i}, F_{2,i}$ and $F_{T,i}$, such that $\xi_{1,i} = \xi_1(F_{1,i}), \xi_{2,i} = \xi_1(F_{2,i})$ and $\xi_{T,i} = \xi_T(F_{T,i})$. After $\mathcal{B}$ provides $\mathcal{A}$ with the $\tau_1+\tau_2+\tau_T = (d_1+1)\cdots(d_n+1)+n+1$ binary strings, $\mathcal{A}$ starts sending queries to $\mathcal{B}$. These queries are either group operation queries, isomorphism queries, or pairing queries.

Group operation: Given a add/subtract selection bit and two operands $\xi_{1,i}, \xi_{1,j}$ compute:

$$F_{1,\tau_1} = F_{1,i} \pm F_{1,j} \in \mathbb{F}_p[X_1,\ldots,X_n].$$

If $F_{1,\tau_1} = F_{1,l}$ for some l that has been already defined, set $\xi_{1,\tau_1} = \xi_{1,l}$. Otherwise, assign ξ_{1,τ_1} a new random string from $E \setminus \{\xi_{1,0},\ldots,\xi_{1,\tau_1-1}\}$. Add $(F_{1,\tau_1}, \xi_{1,\tau_1})$ to L_1, give ξ_{1,τ_1} to $\mathcal{A}$, and increment τ_1. Similar operations are performed for L_2 and L_T.

Isomorphism: Given a string $\xi_{1,i}$ with $0 \leq i < \tau_1$, we let:

$$F_{2,\tau_2} = F_{1,i} \in \mathbb{F}_p[X_1,\ldots,X_n].$$

If $F_{2,\tau_2} = F_{2,l}$ for some l already defined, we set $\xi_{2,\tau_1} = \xi_{2,l}$. Otherwise, we set ξ_{2,τ_2} to a string in $E \setminus \{\xi_{2,0},\ldots,\xi_{2,\tau_2-1}\}$. We add $(F_{2,\tau_2}, \xi_{2,\tau_2})$ to L_2, give ξ_{2,τ_2} to $\mathcal{A}$, and increment τ_2 by one.

Pairing: Given $\xi_{1,i}$ and $\xi_{2,j}$ with i,j already defined, compute:

$$F_{T,\tau_T} = F_{1,i} \cdot F_{2,j} \in \mathbb{F}_p[X_1,\ldots,X_n].$$

If $F_{T,\tau_T} = F_{T,l}$ for some l that has been already defined, set $\xi_{T,\tau_T} = \xi_{T,l}$. Otherwise, assign a new random string, distinct from all of the previous strings. Add $(F_{T,\tau_T}, \xi_{T,\tau_T})$ to L_T, give ξ_{T,τ_T} to $\mathcal{A}$, and increment τ_T.

From the way how the algorithm $\mathcal{B}$ augments the lists, it is obvious that at step τ in the game, $\tau_1+\tau_2+\tau_T \leq \tau + (d_1+1)\ldots(d_n+1)+n+1$, because $\tau+(d_1+1)\ldots(d_n+1)+n+1$ is the maximum possible number of tuples in lists L_1, L_2 and L_T.

After at most q queries to oracles, $\mathcal{A}$ terminates and outputs the following tuple:

$$\left(\xi_1(\chi), \left\{\{S_j^{(i_1,\ldots,i_{j-1})}\}_{i_1\in[d_1],\ldots,i_{j-1}\in[d_{j-1}]}, \{\xi_1(\varphi_j^{(i_1,\ldots,i_{j-1})})\}_{i_1\in[d_1],\ldots,i_{j-1}\in[d_{j-1}]}\right\}_{j\in[1,n]}\right),$$

such that $S_j^{(i_1,\ldots,i_{j-1})} \subseteq \mathbb{F}_p^j$ and the cardinality of $S_j^{(i_1,\ldots,i_{j-1})}$ is d_j+1. There are some $F_{1,i}$ that correspond to $\xi_1\left(\varphi_j^{(i_1,\ldots,i_{j-1})}\right)$. There is also $F_{1,l}$ corresponding to $\xi_1(\chi)$. We define the polynomial $F_{T,*}(X_1,\ldots,X_n)$ as

$$F_{T,*}(X_1,\ldots,X_n) = F_{1,l}(X_1,\ldots,X_n) - \sum_{j=1}^{n} \sum_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} F_{1,i}(X_1,\ldots,X_n) Z_{S_j^{(i_1,\ldots,i_{j-1})}}(X_j),$$

where the first sum in the double sum goes from $j=1$ to n and the second one goes through all polynomials $F_{1,i}$ corresponding to all $\varphi_j^{(i_1,\ldots,i_{j-1})}$.

At this stage, the algorithm $\mathcal{B}$ samples random $(t_1, \ldots, t_n) \in \mathbb{F}_p^n$. If $\mathcal{A}$'s answer is correct, then:

$$F_{T,*}(t_1, \ldots, t_n) = 0, \quad F_{1,l}(t_1, \ldots, t_n) - \sum_{j=2}^{n} \sum_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} F_{1,i}(t_1, \ldots, t_n) Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j) \neq 0 \tag{3}$$

The Equation (3) corresponds to the adversary $\mathcal{A}$ breaking the $\mathsf{GARSDH}(n)$ instance (for $\mathbf{d} = (d_1, \ldots, d_n)$) in the discrete logarithms of the group elements.

There are two possibilities for $\mathcal{A}$ to break the $\mathsf{GARSDH}(n)$ instance. Either $F_{T,*}(X_1, \ldots, X_n) = 0$, or $F_{T,*}(X_1, \ldots, X_n) \neq 0$ and $(t_1, \ldots, t_n)$ is among the zeros of the polynomial $F_{T,*}(X_1, \ldots, X_n)$. The polynomial $F_{T,*}(X_1, \ldots, X_n)$ cannot be the identically zero polynomial, because it would violate the second winning condition from the $\mathsf{GARSDH}(n)$ assumption.

Now the adversary $\mathcal{A}$ may win in two possible ways. Either the simulation provided by $\mathcal{B}$ is not perfect, in which case we can assume that $\mathcal{A}$ won, or the substitution of $(t_1, \ldots, t_n)$ into $(X_1, \ldots X_n)$ satisfies eq. (3). The simulation provided by $\mathcal{B}$ is perfect unless the substitution of $(t_1, \ldots, t_n)$ into $(X_1, \ldots X_n)$ creates an equality relation between the simulated group elements that was not revealed to $\mathcal{A}$, during the query phase. Hence, the success probability of $\mathcal{A}$ is bounded by the probability of one of the following four events happening.

1. $F_{1,i}(t_1, \ldots, t_n) = F_{1,j}(t_1, \ldots, t_n)$ for a pair of polynomials such that $i \neq j$.
2. $F_{2,i}(t_1, \ldots, t_n) = F_{2,j}(t_1, \ldots, t_n)$ for a pair of polynomials such that $i \neq j$.
3. $F_{T,i}(t_1, \ldots, t_n) = F_{T,j}(t_1, \ldots, t_n)$ for a pair of polynomials such that $i \neq j$.
4. $F_{T,*}(t_1, \ldots, t_n) = 0$ and $F_{1,l}(t_1, \ldots, t_n) - \sum_{j=2}^{n} \sum_{i_1,\ldots,i_{j-1}=0}^{d_1,\ldots,d_{j-1}} F_{1,i}(t_1, \ldots, t_n) Z_{S_j^{(i_1,\ldots,i_{j-1})}}(t_j) \neq 0$.

The first three events correspond to the simulation provided by $\mathcal{B}$ to $\mathcal{A}$ not being perfect and the last event corresponds to event when $\mathcal{A}$ succeeds in breaking the $\mathsf{GARSDH}(n)$ instance for $\mathbf{d} = (d_1, \ldots, d_n)$. The probabilities of the first three events are at most

$$\binom{\tau_1}{2} \frac{\sum_{i=1}^{n} d_i}{p^n}, \binom{\tau_2}{2} \frac{\sum_{i=1}^{n} d_i}{p^n}, \binom{\tau_T}{2} \frac{2(\sum_{i=1}^{n} d_i)}{p^n}$$

respectively. This follows from applying the Schwartz-Zippel lemma on the maximum possible degrees of the polynomials $F_{1,i}(X_1, \ldots, X_n), F_{2,i}(X_1, \ldots, X_n)$ and $F_{T,i}(X_1, \ldots, X_n)$. These maximum possible degrees are $\sum_{i=1}^{n} d_i, \sum_{i=1}^{n} d_i$ and $2(\sum_{i=1}^{n} d_i)$ respectively. The probability of the last event happening is at most $(d_1 + \ldots + d_n + d' + 1)/p^n$, where $d_1 + \ldots + d_n + d' + 1$ is the upper bound on the total degree of $F_{T,*}(X_1, \ldots, X_n)$ and $d' = \max\{d_1, \ldots, d_n\}$. We have obtained this upper bounds by applying Schwartz-Zippel lemma on the maximum possible total degree of the polynomial $F_{T,*}(X_1, \ldots, X_n)$.

Hence $\mathcal{A}$ wins the game with probability at most

$$\epsilon \leq \binom{\tau_1}{2} \frac{\sum_{i=1}^{n} d_i}{p^n} + \binom{\tau_2}{2} \frac{\sum_{i=1}^{n} d_i}{p^n} + \binom{\tau_T}{2} \frac{2(\sum_{i=1}^{n} d_i)}{p^n} + \frac{d_1 + \ldots + d_n + d' + 1}{p^n}.$$

Using the same argument as in the proof of $\mathsf{ARSDH}(n)$, we can show the same upper bound on the probability of GGM adversary breaking the $\mathsf{GARSDH}(n)$ assumption:

$$\epsilon \leq \left(q + \prod_{i=1}^{n} (d_i + 1) + n + 1 \right)^2 \frac{\sum_{i=1}^{n} d_i}{p^n} + \frac{\sum_{i=1}^{n} d_i + d'}{p^n}.$$

This concludes the proof. □

Note that the upper bounds on the success probabilities of an $\mathsf{ARSDH}(n)$ adversary and a $\mathsf{GARSDH}(n)$ adversary differ only in the last term $\frac{\sum_{i=1}^{n} d_i + d_n}{p^n}$ respectively $\frac{\sum_{i=1}^{n} d_i + d'}{p^n}$, where d_n is the maximal degree of X_n and d' is the maximal degree of *any* variable. Thus, to some extent, the respective degrees of the variables might affect the hardness of GARSDH, unlike in ARSDH, where the additional term depends only on d_n. In more detail, the two bounds are so similar due to the following two specific points in the proofs:

1. The number of bit strings the generic $\mathsf{ARSDH}(n)$ adversary for $\mathbf{d} = (d_1, \ldots, d_n)$ gets is the same as the number of bit strings the generic $\mathsf{GARSDH}(n)$ for $\mathbf{d} = (d_1, \ldots, d_n)$ adversary gets. So we are using the same upper bound on the number of elements in the lists provided by the algorithm $\mathcal{B}$.
2. The degrees of the polynomials that ensure that the simulation was correct, or that the adversary failed in breaking the $\mathsf{ARSDH}(n)$ assumption are almost the same as the degrees of the same polynomials ensuring correctness of the $\mathsf{GARSDH}(n)$ simulation. So the probabilities of events 1 through 3 are very similar in the two proofs.

9 (Non-)Uniqueness of Proofs of the KZG PCS Family

In this section, we revisit the notion of (computational) uniqueness of proofs for the KZG family. Our motivation is the following. Instead of the extended KZG PoKoP, our multivariate batching lemma (Lemma 2) could be used to analyze the canonical KZG PoKoP if we were able to efficiently extract a π-consistent accepting tree for the canonical KZG PoKoP. Thus, giving a proof of the knowledge-soundness of the canonical KZG PoKoP under the plausibly weaker ARSDH assumption.

As it turns out, the task of extracting π-consistent transcripts for the canonical KZG PoKoP is closely related to the property of computational uniqueness of proofs for polynomial commitment schemes, which ensures that an efficient adversary cannot generate distinct accepting proofs on the same pair of evaluation point $\mathbf{x}$ and value z. The formal definition of this notion, which is orthogonal to evaluation binding or knowledge-soundness, is presented below.

Definition 21 (Computational Uniqueness of Proofs). *We say a* multivariate polynomial commitment scheme $\mathsf{PCS} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ has computationally unique proofs *w.r.t.* PGen *if, for all degree bounds* $\mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$ *and* PPT *adversaries* $\mathcal{A}$*, it holds that*

$$\Pr\left[\begin{array}{l|l} \pi \neq \pi', & \mathsf{aux} \leftarrow \mathsf{PGen}(\lambda), \\ \mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi) = 1, & \mathsf{ck} \leftarrow \mathsf{KGen}(1^\lambda, \mathbf{d}, \mathsf{aux}), \\ \mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi') = 1 & (C, \mathbf{x}, z, \pi, \pi') \leftarrow \mathcal{A}(\mathsf{ck}) \end{array}\right] \in \mathsf{negl}(\lambda).$$

Note that the computational uniqueness of proofs for the multivariate KZG scheme is a necessary condition to establishing that any efficiently extractable tree of accepting transcripts for the canonical KZG PoKoP is π-consistent. Next, we discuss the KZG PCS family from the perspective of uniqueness of proofs. First, we show that even though the univariate KZG proofs are not unique the univariate scheme has computational uniqueness of proofs under SBDH. Then, we show that, already in the bivariate setting, the computational uniqueness of proofs strongly does not hold as it is possible to rerandomize the evaluation proofs. Even though it hinders the applicability of our proof technique based on ARSDH, the possibility of rerandomizing evaluation proofs for multivariate KZG might be of independent interest in the future.

Computational uniqueness of proofs for univariate KZG. In this subsection, we use n to correspond to the degree of a univariate polynomial. Below, we formally recall the standard SBDH assumption.

Definition 22 (SBDH Assumption). *For* $n \in \mathbb{N}$ *and a bilinear group generator* PGen*, we say that the* n-SBDH *assumption holds for* PGen *if, for all* PPT *adversaries* $\mathcal{A}$*, it holds that*

$$\Pr\left[\begin{array}{l|l} \gamma + c \neq 0, & \mathsf{gp} = (p, [1]_1, [1]_2, \bullet) \leftarrow \mathsf{PGen}(\lambda), \gamma \leftarrow \mathbb{F}_p, \\ [\varphi]_1 = \left[\dfrac{1}{\gamma + c}\right]_1 & (c, [\varphi]_1) \leftarrow \mathcal{A}(\mathsf{gp}, ([\gamma^i]_1)_{i=1}^n, [\gamma]_2) \end{array}\right] \in \mathsf{negl}(\lambda).$$

We say the SBDH *assumption holds for* PGen *if the* n-SBDH *assumption holds for* PGen *for all* $n \in \mathsf{poly}(\lambda)$.

The univariate KZG scheme is the special case of the multivariate KZG scheme (Figure 2) that we recall in Figure 10.

$\mathsf{KZG.KGen}(1^\lambda, n, \mathsf{aux} = \mathsf{gp} = (p, [1]_1, [1]_2, \bullet))$:
Sample $\sigma \leftarrow \mathbb{F}_p$.
Output $\mathsf{ck} = \left(\mathsf{gp}, \left([\sigma^i]_1\right)_{i=1}^n, [\sigma]_2\right)$.
$\mathsf{KZG.Com}(\mathsf{ck}, f)$:
Output $C = [f(\sigma)]_1$.
$\mathsf{KZG.Open}(\mathsf{ck}, f, x)$:
Compute $z = f(x)$ and $\pi = \left[\frac{f(X)-z}{\sigma-x}\right]_1$.
Output (z, π).
$\mathsf{KZG.Ver}(\mathsf{ck}, C, x, z, \pi)$:
If $(C - [z]_1) \bullet [1]_2 = \pi \bullet [\sigma - x]_2$ then output 1, and 0 otherwise.

Fig. 10. The univariate KZG PCS defined w.r.t. bilinear group parameters gp.

Lemma 11. *If the n-SBDH assumption holds for PGen then the univariate KZG scheme (Figure 10) has computational uniqueness of proofs w.r.t. PGen when used with degree bound n.*

Proof. We show how any adversary breaking the computational uniqueness of univariate KZG could be used to break the n-SBDH assumption.

Let $\mathsf{srs} = (([\sigma^i]_1)_{i=0}^n, [1, \sigma]_2)$ be the challenge we are given in n-SBDH and let $\mathcal{A}$ be the PPT adversary that can break the computational uniqueness of KZG with non-negligible probability. We call $\mathcal{A}$ on the srs as the commitment key and we get back a tuple (C, x, z, π, π') that should satisfy

$$(C - [z]_1) \bullet [1]_2 = \pi \bullet [\sigma - x]_2,$$
$$(C - [z]_1) \bullet [1]_2 = \pi' \bullet [\sigma - x]_2.$$

We set $\pi = [p]_1, \pi' = [p']_1$. By multiplying the first equation by the inverse of the second we get

$$[0]_T = [p - p']_1 \bullet [\sigma - x]_2, \tag{18}$$
$$[0]_T = ([1]_1 \bullet [1]_2)^{(\sigma-x)(p-p')}.$$

The only way this equation can be satisfied is if the order of the element g_T denoted as $ord(g_T)$ divides $(\sigma - x)(p - p')$. We know that if the target group has prime order q, then every element in the group has the same prime order q and so $ord(g_T) = q$.

To satisfy $q \mid (\sigma - x)(p - p')$, either $q|(p - p')$ or $q|(\sigma - x)$. In the first case, since both p and p' come from the field $\mathbb{F}_q$, the only way q could divide $p - p'$ is only if $p - p' \equiv 0 \ mod \ q$. If $p' = p + k \cdot q$ for some $k \in \mathbb{Z}$, then we would have $\pi' = [p']_1 = [p + kq]_1 = [p]_1 = \pi$. This happens only with a negligible probability as we assume that $\mathcal{A}$ can break the computational uniqueness with a non-negligible probability.

Now, if q divides $(\sigma - x)$ we have $\sigma - x = 0$ as σ is chosen randomly from $\mathbb{F}_q$. Thus, we know the value of $\sigma = x$ and we can compute $[\varphi]_1 = [\frac{1}{\sigma-x}]_1$. We can set $c = -x$ and output c and $[\varphi]_1$, breaking the n-SBDH instance. □

Note that the above proof would be simpler under the somewhat stronger SBDH* assumption where the adversary's winning condition is described in the target group (Definition 25). Under SBDH*, we could conclude the proof after eq. (18) since we obtain

$$([1]_1 \bullet [1]_2)^{\frac{1}{\sigma-x}} = [p - p']_1 \bullet [1]_2,$$

where the left side already satisfies the winning condition in the target group, and the right side is easily computable. However, establishing the computational uniqueness of proofs under SBDH* assumption is, in some sense, weaker, as it allows the adversary to exploit the pairing to construct elements in the target group that it might not be able to produce in the source groups.

Non-uniqueness of proofs for multivariate KZG. Next, we show that the computational uniqueness of proofs does not hold for the bivariate KZG scheme. Below, we demonstrate that, for any evaluation point, it is possible to construct exponentially many distinct accepting proofs.

Theorem 13. *Let* $\mathsf{ck} \leftarrow \mathsf{KZG.KGen}(1^\lambda, \mathbf{d} = (m, n), \mathsf{gp})$ *be a* KZG *commitment key for some* $\mathsf{gp} = (p, [1]_1, [1]_2, \bullet)$ *and* $\mathbf{d} = (m, n) \in \mathbb{N}^2$. *If* $\mathsf{Ver}(\mathsf{ck}, C, (x, y), z, \pi = (\pi_1, \pi_2)) = 1$ *for some* $C, \pi_1, \pi_2 \in \mathbb{G}_1$ *and* $x, y, z \in \mathbb{F}_p$ *then*

$$|\{\pi' \in \mathbb{G}_1 \times \mathbb{G}_1 \mid \mathsf{Ver}(\mathsf{ck}, C, (x, y), z, \pi') = 1\}| \geq |\mathbb{G}_1|.$$

Proof. Let us have a transcript $(\mathsf{ck}, C, (x, y), z, \pi)$ such that $\mathsf{Ver}(\mathsf{ck}, C, (x, y), z, \pi) = 1$. We show how to construct a new proof π' such that $\mathsf{Ver}(\mathsf{ck}, C, (x, y), z, \pi') = 1$. Let $\pi = (\pi_1, \pi_2)$ where $\pi_1, \pi_2 \in \mathbb{G}_1$ and let us have arbitrary $\mu_1, \mu_2 \in \mathbb{G}_1$. Then we can shift the original proof by these values, creating a new proof $\pi' = (\pi_1', \pi_2') = (\pi_1 + \mu_1, \pi_2 + \mu_2)$.

For this to be a valid proof, we have to have

$$(C - [z]_1) \bullet [1]_2 = \pi_1' \bullet [\sigma - x]_2 \otimes \pi_2' \bullet [\tau - y]_2.$$

this can be rewritten as

$$\begin{aligned}
(C - [z]_1) \bullet [1]_2 &= (\pi_1 + \mu_1) \bullet [\sigma - x]_2 \otimes (\pi_2 + \mu_2) \bullet [\tau - y]_2, \\
(C - [z]_1) \bullet [1]_2 &= \pi_1 \bullet [\sigma - x]_2 \otimes \pi_2 \bullet [\tau - y]_2 \otimes \mu_1 \bullet [\sigma - x]_2 \otimes \mu_2 \bullet [\tau - y]_2, \\
(C - [z]_1) \bullet [1]_2 &= (C - [z]_1) \bullet [1]_2 \otimes \mu_1 \bullet [\sigma - x]_2 \otimes \mu_2 \bullet [\tau - y]_2.
\end{aligned}$$

When we set $\mu_1 = [\tilde{\mu_1}]_1$ and $\mu_2 = [\tilde{\mu_2}]_1$ we can derive the following

$$\begin{aligned}
[\tilde{\mu_1}]_1 \bullet [x - \sigma]_2 &= [\tilde{\mu_2}]_1 \bullet [\tau - y]_2, \\
\tilde{\mu_1}(x - \sigma) &\equiv \tilde{\mu_2}(\tau - y) \mod p, \\
\tilde{\mu_1} &\equiv \tilde{\mu_2} \frac{\tau - y}{x - \sigma} \mod p.
\end{aligned}$$

From this we can see that for every $\mu_1 \in \mathbb{G}_1$ there is a $\mu_2 \in \mathbb{G}_1$ such that the shifted proof $\pi' = (\pi_1 + \mu_1, \pi_2 + \mu_2)$ is also a valid proof. Thus, we have at least $|\mathbb{G}_1|$ different proofs that are accepted by the verifier. □

The above result demonstrates that the KZG scheme defined in Figure 2 does not satisfy the computational uniqueness of proofs, as uniqueness fails already in the bivariate case. Moreover, to see that computational uniqueness does not hold for an arbitrary number of variables $n > 2$, we can use our discussion about the randomized multivariate KZG scheme in Section 7.3. Specifically, using the quotient polynomials $\widetilde{q_i}$ defined from the standard quotient polynomials q_i by an arbitrary vector α, one can construct multiple distinct yet valid proofs, reinforcing the conclusion that computational uniqueness strongly does not hold for the multivariate scheme.

10 Evaluation Binding of the KZG PCS Family

The multivariate KZG scheme from [PST13] was introduced as a multivariate polynomial evaluation scheme in the context of "signatures of correct computation." In fact, Papamanthou et al. do not discuss their scheme's standard evaluation binding property but, instead, prove the selective security of their multivariate polynomial evaluation scheme. In this section, we present their results in the language of evaluation binding for polynomial schemes since translating between standard evaluation binding for polynomial commitment schemes and their notion of selective (resp. adaptive) security for multivariate polynomial evaluation schemes is not completely straightforward.

Recall that evaluation binding ensures that an adversary cannot generate two distinct yet valid opening proofs for a committed polynomial at a single evaluation point. Intuitively, this guarantees that the adversary behaves consistently with a function. The formal definition of standard evaluation binding, which has been previously studied for various polynomial commitment schemes, is presented below.

Definition 23. *We say a* multivariate polynomial commitment scheme $\mathsf{PCS} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ is evaluation binding *w.r.t.* PGen *if, for all degree bounds* $\mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$ *and* PPT *adversaries* $\mathcal{A}$*, it holds that*

$$\Pr\left[\begin{array}{l|l} z \neq z', & \mathsf{aux} \leftarrow \mathsf{PGen}(\lambda), \\ \mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi) = 1, & \mathsf{ck} \leftarrow \mathsf{KGen}(1^\lambda, \mathbf{d}, \mathsf{aux}), \\ \mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z', \pi') = 1 & (C, \mathbf{x}, z, \pi, z', \pi') \leftarrow \mathcal{A}(\mathsf{ck}) \end{array}\right] \in \mathsf{negl}(\lambda).$$

Evaluation binding for univariate KZG supporting polynomials of degree at most d has been proven under the d-SBDH assumption [KZG10]. However, proving evaluation binding for the multivariate KZG presents additional challenges. In the language of signatures of correct computation, evaluation binding corresponds to *adaptive* security, whereas [PST13] proved only the weaker notion of *selective* security for the multivariate KZG scheme. Selective security for multivariate polynomial evaluation schemes corresponds to a relaxation of the standard evaluation binding that we call *evaluation binding on average* and define below.

Definition 24. *We say a* multivariate polynomial commitment scheme $\mathsf{PCS} = (\mathsf{KGen}, \mathsf{Com}, \mathsf{Open}, \mathsf{Ver})$ is evaluation binding on average *w.r.t.* PGen *if, for any* $\mathbf{d} = (d_1, \ldots, d_n) \in \mathsf{poly}(\lambda)$ *and for all* PPT *adversaries* $\mathcal{A}$*, it holds that*

$$\Pr\left[\begin{array}{l|l} z \neq z', & \mathsf{aux} \leftarrow \mathsf{PGen}(\lambda), \\ \mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z, \pi) = 1, & \mathsf{ck} \leftarrow \mathsf{KGen}(1^\lambda, \mathbf{d}, \mathsf{aux}), \\ \mathsf{Ver}(\mathsf{ck}, C, \mathbf{x}, z', \pi') = 1 & \mathbf{x} \leftarrow \mathbb{F}_p^n, \\ & (C, z, \pi, z', \pi') \leftarrow \mathcal{A}(\mathsf{ck}, \mathbf{x}) \end{array}\right] \in \mathsf{negl}(\lambda).$$

The above relaxation effectively means that, while there might exist specific evaluation points where an adversary could violate evaluation binding, the probability of breaking the property over a randomly chosen evaluation point remains negligible. Note that this definition, though slightly weaker, might be sufficient for applications such as SNARKs, where we only need to evaluate polynomials at random points. The proof of evaluation binding on average for the multivariate KZG scheme from [PST13] relies on a variant of the SBDH assumption with the winning identity defined in the target group, which we recall next.

Definition 25 (SBDH^* Assumption). *For* $n \in \mathbb{N}$ *and a bilinear group generator* PGen*, we say that the* n-SBDH^* *assumption holds for* PGen *if, for all* PPT *adversaries* $\mathcal{A}$*, it holds that*

$$\Pr\left[\begin{array}{l|l} \gamma + c \neq 0, & \mathsf{gp} = (p, [1]_1, [1]_2, \bullet) \leftarrow \mathsf{PGen}(\lambda), \gamma \leftarrow \mathbb{Z}_p, \\ [\varphi]_T = ([1]_1 \bullet [1]_2)^{\frac{1}{\gamma + c}} & (c, [\varphi]_T) \leftarrow \mathcal{A}(\mathsf{gp}, ([\gamma^i]_1)_{i=1}^n, [\gamma]_2) \end{array}\right] \in \mathsf{negl}(\lambda).$$

We say the SBDH^* *assumption holds for* PGen *if the* n-SBDH^* *assumption holds for* PGen *for all* $n \in \mathsf{poly}(\lambda)$*.*

Next, we illustrate the proof of evaluation binding on average from [PST13] on the bivariate variant of KZG.

Theorem 14. *Under the* $(m+n)$-SBDH^* *assumption, the multivariate polynomial commitment scheme* KZG *presented in Figure 2 used with degree bounds* $\mathbf{d} = (m, n)$ *is evaluation binding on average.*

Proof. We show that any adversary $\mathcal{A}$ breaking the evaluation binding property of the KZG polynomial commitment scheme when used with degree bounds $\mathbf{d} = (m, n)$ can be used to break the $(m+n)$-SBDH^* assumption. Given an $(m+n)$-SBDH^* challenge $\left([1]_1, [\gamma]_1, [\gamma^2]_1, \ldots, [\gamma^{n+m}]_1\right)$, where γ is chosen uniformly randomly from $\mathbb{F}_p^*$, our goal is to output a pair $(c, \varphi) \in (\mathbb{F}_p, \mathbb{G}_1)$ such that $c \neq -\gamma$ and $\varphi = ([1]_1 \bullet [1]_2)^{\frac{1}{\gamma + c}}$.

First set $\sigma = \gamma$ and sample $(x, y) \leftarrow \mathbb{F}_p^2$ and $r \leftarrow \mathbb{F}_p$. Then compute $s = y - r$ and set $\tau = r\gamma + s$. In this way, we have set up the secret evaluation point for the commitment key (σ, τ) and the uniform evaluation point (x, y) such that both $\sigma - x = \gamma - x$ and $\tau - y = r(\gamma - x)$ are divisible by $\gamma - x$. Now, we compute the KZG commitment key ck with respect to (σ, τ) and degree bounds (m, n), which we can do because the $(m+n)$-SBDH^* instance contains the values $[\gamma^i]_1$ for $i = 0, \ldots, n+m$, and $n+m$ is exactly the highest degree that we need to compute to create the bivariate commitment key.

Next, we argue that even though we have created a correlated pair of a commitment key ck and evaluation point (x, y), its distribution is identical to $\mathcal{A}$'s input in the evaluation binding experiment in Definition 24. Since (x, y) is a uniformly random evaluation point and $\sigma = \gamma$ is also uniformly random, it remains to be shown that $\tau = r\gamma + s = r\gamma + y - r = r(\gamma - 1) + y$ is also uniformly random and independent of (x, y) and σ. To this end, consider the function $\varphi_b : \mathbb{F}_p \to \mathbb{F}_p$, $\varphi_b : r \mapsto rb$. It is easy to see that φ_b is a bijection for all $0 \neq b \in \mathbb{N}$, since all non-zero b have a multiplicative inverse in $\mathbb{F}_p$. Thus, if $\gamma \neq 1$ then τ is distributed uniformly independently of y. On the other hand, $\gamma = 1$ with probability exactly p^{-1}, which is exponentially small in the security parameter.

Thus, $\mathcal{A}(\mathsf{ck}, (x, y))$ returns a tuple (C, z, z', π, π') that breaks the evaluation binding with a non-negligible probability, i.e., if we denote $\pi = (\pi_1, \pi_2)$ and $\pi' = (\pi'_1, \pi'_2)$, we get that

$$(C - [z]_1) \bullet [1]_2 = \pi_1 \bullet [\sigma - x]_2 \otimes \pi_2 \bullet [\tau - y]_2,$$
$$(C - [z']_1) \bullet [1]_2 = \pi'_1 \bullet [\sigma - x]_2 \otimes \pi'_2 \bullet [\tau - y]_2.$$

After rewriting $\pi_1 = [p_1]_1, \pi_2 = [p_2]_1, \pi'_1 = [p'_1]_1, \pi'_2 = [p'_2]_1$ and multiplying the first equation by the inverse of the second, we get

$$[z' - z]_1 \bullet [1]_2 = [p_1 - p'_1]_1 \bullet [\sigma - x]_2 \otimes [p_2 - p'_2]_1 \bullet [\tau - y]_2.$$

If we set $\delta = z' - z \neq 0$ and raise both sides of the equations to $\frac{1}{\delta(\gamma - x)}$, we get that

$$([1]_1 \bullet [1]_2)^{\frac{1}{\gamma - x}} = ([p_1 - p'_1]_1 \bullet [1]_2 \otimes [p_2 - p'_2]_1 \bullet [r]_2)^{\frac{1}{\delta}}.$$

Notice that, on the left side, we have the desired $([1]_1 \bullet [1]_2)^{\frac{1}{\gamma + c}}$ for $c = -x$, while we can compute the element on the right side of the equality from the tuple $(C, z, z', (\pi_1, \pi_2), (\pi'_1, \pi'_2))$ we got from $\mathcal{A}$. Thus, we can break the $(m + n)$-SBDH^* instance by outputting $(x, ([p_1 - p'_1] \bullet [1]_2 \otimes [p_2 - p'_2] \bullet [r]_2)^{\frac{1}{\delta}})$. □

Generally, when dealing with an n-variate polynomial commitment scheme where the verification check is of the form

$$(C - [z]_1) \bullet [1]_2 = \bigotimes_{i=1}^{n} \pi_i \bullet [t_i - x_i]_2$$

with trapdoors t_i, we can apply the same trick to derive the relation

$$([1]_1 \bullet [1]_2)^{\frac{1}{\gamma - x_1}} = \left(\bigotimes_{i=1}^{n} [p_i - p'_i]_1 \bullet [r_i]_2 \right)^{\frac{1}{\delta}}.$$

To achieve this, we must choose t_i such that $t_i - x_i = r_i(\gamma - x_1)$.

The key observation is that the trick works by setting the trapdoors t_i such that we can later divide by $t_1 - x_1$. However, this dependency prevents us from applying this proof to the standard definition of evaluation binding, where the evaluation point x is not given in advance. In [PST13], this issue is addressed by introducing the randomized multivariate KZG scheme we discussed in Section 7.3. The randomization of the KZG proof enables again the selection of random evaluation points, ensuring the divisibility by $t_1 - x_1$. Therefore, for the randomized KZG scheme, [PST13] proved the standard evaluation binding as presented in Definition 23.

Acknowledgments

Juraj Belohorec, Pavel Hubáček, and Kristýna Mašková were partially suported by the Academy of Sciences of the Czech Republic (RVO 67985840), Czech Science Foundation GAČR grant No. 25-16311S, and by Zircuit. Pavel Dvořák was supported by Czech Science Foundation GAČR grant No. 22-14872O.

References

ACK21. Thomas Attema, Ronald Cramer, and Lisa Kohl. A compressed Σ-protocol theory for lattices. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part II*, volume 12826 of *Lecture Notes in Computer Science*, pages 549–579. Springer, 2021.

AM14. Aysajan Abidin and Aikaterini Mitrokotsa. Security aspects of privacy-preserving biometric authentication based on ideal lattices and ring-lwe. In *2014 IEEE International Workshop on Information Forensics and Security, WIFS 2014, Atlanta, GA, USA, December 3-5, 2014*, pages 60–65, 2014.

BB04. Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan L. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, pages 56–73, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

BDFG21. Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Halo infinite: Proof-carrying data from additive polynomial commitments. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part I*, volume 12825 of *Lecture Notes in Computer Science*, pages 649–680. Springer, 2021.

BFS19. Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent SNARKs from DARK compilers. *IACR Cryptol. ePrint Arch.*, page 1229, 2019.

BMM+24. Anubhav Baweja, Pratyush Mishra, Tushar Mopuri, Karan Newatia, and Steve Wang. Scribe: Low-memory SNARKs via read-write streaming. *IACR Cryptol. ePrint Arch.*, page 1970, 2024.

CBBZ23. Binyi Chen, Benedikt Bünz, Dan Boneh, and Zhenfei Zhang. HyperPlonk: Plonk with linear-time prover and high-degree custom gates. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part II*, volume 14005 of *Lecture Notes in Computer Science*, pages 499–530. Springer, 2023.

CGG+23. Matteo Campanelli, Nicolas Gailly, Rosario Gennaro, Philipp Jovanovic, Mara Mihali, and Justin Thaler. Testudo: Linear time prover SNARKs with constant size proofs and square root size universal setup. *IACR Cryptol. ePrint Arch.*, page 961, 2023.

CHM+20. Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Psi Vesely, and Nicholas P. Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 738–768. Springer, 2020.

DMS24. Michel Dellepere, Pratyush Mishra, and Alireza Shirzad. Garuda and pari: Faster and smaller SNARKs via equifficient polynomial commitments. *IACR Cryptol. ePrint Arch.*, page 1245, 2024.

FKL18. Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 33–62. Springer, 2018.

FS86. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Annual International Cryptology Conference*, 1986.

GR19. Alonso González and Carla Ràfols. Shorter pairing-based arguments under standard assumptions. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part III*, volume 11923 of *Lecture Notes in Computer Science*, pages 728–757. Springer, 2019.

GWC19. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. *IACR Cryptol. ePrint Arch.*, page 953, 2019.

KZG10. Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 177–194. Springer, 2010.

Lee21. Jonathan Lee. Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part II*, pages 1–34, 2021.

Lib24. Benoît Libert. Simulation-extractable KZG polynomial commitments and applications to hyperplonk. In *Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part II*, pages 68–98, 2024.

Lin03. Yehuda Lindell. Parallel coin-tossing and constant-round secure two-party computation. *J. Cryptol.*, 16(3):143–184, 2003.

LLZ+24. Chongrong Li, Yun Li, Pengfei Zhu, Wenjie Qu, and Jiaheng Zhang. HyperPianist: Pianist with linear-time prover via fully distributed HyperPlonk. *IACR Cryptol. ePrint Arch.*, page 1273, 2024.

LPS23. Helger Lipmaa, Roberto Parisella, and Janno Siim. Algebraic group model with oblivious sampling. In Guy N. Rothblum and Hoeteck Wee, editors, *Theory of Cryptography - 21st International Conference, TCC 2023, Taipei, Taiwan, November 29 - December 2, 2023, Proceedings, Part IV*, volume 14372 of *Lecture Notes in Computer Science*, pages 363–392. Springer, 2023.

LPS24a. Helger Lipmaa, Roberto Parisella, and Janno Siim. Constant-size zk-SNARKs in ROM from falsifiable assumptions. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI*, volume 14656 of *Lecture Notes in Computer Science*, pages 34–64. Springer, 2024.

LPS24b. Helger Lipmaa, Roberto Parisella, and Janno Siim. On knowledge-soundness of Plonk in ROM from falsifiable assumptions. *IACR Cryptol. ePrint Arch.*, page 994, 2024.

LXZ+24. Tianyi Liu, Tiancheng Xie, Jiaheng Zhang, Dawn Song, and Yupeng Zhang. Pianist: Scalable zkRollups via fully distributed zero-knowledge proofs. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pages 1777–1793. IEEE, 2024.

LZW+24. Xuanming Liu, Zhelei Zhou, Yinghao Wang, Bingsheng Zhang, and Xiaohu Yang. Scalable collaborative zk-SNARK: Fully distributed proof generation and malicious security. *IACR Cryptol. ePrint Arch.*, page 143, 2024.

MBKM19. Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, pages 2111–2128. ACM, 2019.

Nao03. Moni Naor. On cryptographic assumptions and challenges. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 96–109. Springer, 2003.

PP24. Christodoulos Pappas and Dimitrios Papadopoulos. Sparrow: Space-efficient zkSNARK for data-parallel circuits and applications to zero-knowledge decision trees. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*, pages 3110–3124. ACM, 2024.

PST13. Charalampos Papamanthou, Elaine Shi, and Roberto Tamassia. Signatures of correct computation. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 222–242. Springer, 2013.

Set20. Srinath T. V. Setty. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, pages 704–737, 2020.

Sho97. Victor Shoup. Lower bounds for discrete logarithms and related problems. In *International Conference on the Theory and Application of Cryptographic Techniques*, 1997.

WHZ24. Wenxuan Wu, Soamar Homsi, and Yupeng Zhang. Confidential and verifiable machine learning delegations on the cloud. In *Computer Security - ESORICS 2024 - 29th European Symposium on Research in Computer Security, Bydgoszcz, Poland, September 16-20, 2024, Proceedings, Part II*, pages 182–201, 2024.

WTS+18. Riad S. Wahby, Ioanna Tzialla, Abhi Shelat, Justin Thaler, and Michael Walfish. Doubly-efficient zk-SNARKs without trusted setup. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 926–943. IEEE Computer Society, 2018.

WZXP22. Qiang Wang, Fucai Zhou, Jian Xu, and Su Peng. Tag-based verifiable delegated set intersection over outsourced private datasets. *IEEE Trans. Cloud Comput.*, 10(2):1201–1214, 2022.

ZGK+17a. Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. vSQL: Verifying arbitrary SQL queries over dynamic outsourced databases. *IACR Cryptol. ePrint Arch.*, page 1145, 2017.

ZGK+17b. Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. A zero-knowledge version of vSQL. *IACR Cryptol. ePrint Arch.*, page 1146, 2017.

ZGK+18. Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. vRAM: Faster verifiable RAM with program-independent preprocessing. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 908–925, 2018.

Zha22. Mark Zhandry. To label, or not to label (in generic groups). In *Advances in Cryptology – CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part III*, page 66–96, Berlin, Heidelberg, 2022. Springer-Verlag.