Proof-Carrying Data via Holography Accumulation

Nikitas Paslis[1], Carla Ràfols[1], and Alexandros Zacharakis[2]

[1] Universitat Pompeu Fabra
{nikitas.paslis, carla.rafols}@upf.edu
[2] Hasso Plattner Institute, University of Potsdam
alexandros.zacharakis@hpi.de

Abstract. Succinct non-interactive arguments of knowledge (SNARKs) enable the verification of complex computations via short proofs. Recursive proof composition allows long-running or distributed computations to be verified incrementally, but existing approaches exhibit a fundamental trade-off. Folding-based schemes achieve highly efficient recursion but require provers to maintain and communicate large private state, while stateless approaches such as full SNARK recursion and atomic accumulation incur higher prover costs due to the need to produce and verify a full SNARK proof at each step. We introduce holography accumulation, a framework for stateless recursive proving for SNARKs based on the lincheck or checkable subspace arguments. These SNARKs admit a natural decomposition of verification into witness-dependent checks and public polynomial evaluations encoding the computation. We show that the latter, which we call holographic checks, can be accumulated efficiently across recursive steps. To formalize this idea, we introduce generalized bilinear forms (GBF), a linear-algebraic abstraction capturing the holographic verification procedures of several modern SNARKs. Using this abstraction, we construct generic PCD schemes compatible with both univariate and multivariate polynomial commitment schemes, and present an efficient decider that collapses the accumulated checks to a single polynomial evaluation.

Table of Contents

Proof-Carrying Data via Holography Accumulation .....
Nikitas Paslis, Carla Ràfols, and Alexandros Zacharakis

1 Introduction 1
- 1.1 Our Contributions 2
- 1.2 Further Comparison 3

2 Techniques 4
- 2.1 Decoupling Structure and Witness 5
- 2.2 Generalized Bilinear Forms 8
- 2.3 Proof-Carrying Data Overview 10

3 Definitions and Building Blocks 13

4 Generalized Bilinear Forms 16
- 4.1 Π_{GBF1} 17
- 4.2 Π_{GBF2} 22
- 4.3 Specializations 24

5 Proof Carrying Data 27
- 5.1 $\mathsf{Barebones}_{\nu}$ 27
- 5.2 Accumulation 29
- 5.3 Deciding Non-Uniform PCD 30

A Deferred Definitions and Theorems 34
- A.1 Polynomial Commitments 34
- A.2 Arguments and Accumulators 34
- A.3 Proof-Carrying Data 36

1 Introduction

Succinct Non-interactive ARguments of Knowledge (SNARKs) are compact cryptographic proofs that attest to the correctness of a computation. A fundamental limitation of monolithic SNARKs, however, is that proof generation can only begin once the computation has fully concluded. Consequently, proving very large computations can impose prohibitive memory requirements. Furthermore, monolithic SNARKs certify only a fixed, finite computation, whereas many applications involve computations whose length is not known in advance or evolves over time. Addressing these resource and structural constraints requires *recursive composition*: a technique that mitigates memory overhead by incrementally producing proofs that attest not only to the correctness of a computation step but also to the validity of prior proofs, while keeping verification complexity small.

Incrementally verifiable computation (IVC) [Val08] and its generalization *proof-carrying data* (PCD) [CT10] formalize this setting by enabling parties in a distributed computation, organized over a directed acyclic graph, to locally compute proofs whose production and verification costs are independent of the graph's depth. More broadly, recursion enables long-running programs whose arbitrarily long execution is compressed into a single proof. These capabilities enable numerous theoretical and practical applications, including complexity-preserving [BCCT13, BCTV14] and low-memory [NDC+24] succinct arguments, verifiable MapReduce computations [CTV15], and succinct consensus protocols and blockchains (e.g., recursive rollups and zkVM systems) [BMRS20, CCDW20, BCG24].

Such applications have served as a motivating factor for the design of IVC and PCD schemes with different flavors [BGH19, BCMS20, BCL+21, BDFG21, KST22, KS22, EG23, BC25, BMNW25, BHKZ25]. Such constructions fall into three broad categories: full SNARK recursion, atomic accumulation, and split accumulation (or folding). We describe how a PCD node's prover behaves in each case.

Full SNARK Recursion. The prover receives as input SNARK proofs from the incoming edges and produces a proof attesting to the following: (1) correct execution of its part of the computation, and (2) validity of the incoming SNARK proofs. The latter requirement is realized by including the SNARK's verification logic in the proven statement. An important requirement to achieve recursive proof composition in this way is that the verifier's circuit be sublinear in the size of the computation. This already excludes SNARKs that are only proof-succinct but not verifier-succinct, such as the ones based on inner-product arguments [BCC+16, BBB+18]. Moreover, the class of pairing-based SNARKs [CHM+20, CFF+21, RZ21, GWC19] is excluded as well, since arithmetizing pairing operations either blows up the circuit size due to foreign field arithmetic or requires cycles of pairing-friendly elliptic curves that are rare [CCW19, BMUS23] and inefficient. Therefore, such constructions are based primarily on hash-based SNARKs [COS20, Pol22].

Atomic Accumulation. This category originates from Halo [BGH19] and was formalized in [BCMS20] as atomic accumulation and in [BDFG21] as a public aggregation scheme. Its goal is to lift the restrictions of full SNARK recursion by allowing the prover to verifiably postpone the verifier's expensive checks that cannot be succinctly arithmetized. Here, the prover's inputs are SNARK proofs and "accumulators" which capture the postponed verifier's checks. The prover then combines the accumulators with the expensive-to-verify parts of the SNARK proofs to create a new accumulator. Afterwards, it produces a SNARK proof attesting to the following: (1) the computations of the node were performed correctly, and (2) the accumulation procedure passes verification. Importantly, the accumulation verifier must admit a smaller arithmetization than the SNARK verifier. In the literature, atomic accumulation schemes focus primarily on accumulating polynomial commitment evaluation proofs by leveraging the homomorphic properties of the underlying polynomial commitment scheme.

Split Accumulation/Folding. The final category, introduced by [BCL+21, KST22], takes the approach of postponing verification to the extreme: if two witnesses satisfy an algebraic constraint, then their linear

combination satisfies a "relaxed" version of the constraint. This relaxation comes in the form of allowing "error terms" in the algebraic equation whose structure is predetermined by the constraint itself. Hence, the prover does not create a full SNARK proof. Instead, on input the previous step's witnesses and error terms for the relaxed relation, it first "folds" them via a random linear combination into a new witness and new error term. Then, it computes a witness for an augmented circuit which includes (1) the node's computation, and (2) the verification of the folding procedure. Constructions in this category [BCL+21, KST22, KS22, BC23, KS24, BC25, BHKZ25] rely on homomorphic polynomial commitment schemes to achieve the most efficient PCD schemes, due to the fact that the folding verifier is small (typically a handful of linear combinations of group elements) and that the prover is not producing a full SNARK proof for the augmented circuit. A recent line of work [BMNW24, BMNW25, BCFW26] has explored emulating the homomorphic property using hash-based commitment schemes to construct folding protocols that are plausibly post-quantum secure. Compared to full recursion and atomic accumulation, folding admits PCD schemes that are orders of magnitude more efficient.

Stateful vs Stateless Prover. The performance advantage of split accumulation/ folding over the first two approaches stems from the fact that the prover is *stateful*, in the sense that it requires the previous step's state (i.e. the witness) to proceed with the computation. This leads to two major drawbacks of folding schemes: (1) the communication complexity between provers is high and sometimes the bottleneck of the PCD scheme, and (2) they cannot be easily applied to distributed computing settings where each prover has some secret data that it wants to keep private, unless an expensive blinding phase is run in each step. Full recursion and atomic accumulation, on the other hand, are *stateless*: the step's prover only needs to access succinct public information. However, both approaches still require the prover to compute at least a full SNARK proof for the augmented circuit at each step. This leads us to the central question of this work.

Can we bridge the efficiency gap between stateful and stateless recursion?

We answer the question affirmatively for a particular class of SNARK constructions, namely those that rely on the so-called "lincheck argument" [BCR+19] or "checkable subspace argument" (CSS) [RZ21]. Schemes of this class [CHM+20, COS20, Set20, CFF+21, RZ21, STW23] reduce verifying a computation to (1) correctness of witness-related polynomial commitment evaluations, and (2) correctness of public polynomial evaluations encoding the computation. We refer to the latter checks – performed on polynomials that encode the structure of the computation and do not depend on the witness – as *holographic checks*.

While proving the first component is strictly necessary for proof succinctness, proving the second is required only for verification succinctness. Crucially, the second component depends solely on the structure of the constraint system (i.e. the sparse matrix encoding in [CHM+20] and the sparse polynomial evaluation in [Set20]) and typically dominates the prover's overhead. A key observation, also noted in [BHKZ25], is that in a recursive setting, the correctness of these "computation encoding" polynomials need not be proven via a SNARK at every step; instead, they can be efficiently accumulated. Since these checks involve only short public information (verifier randomness and claimed evaluations), accumulating them does not compromise the stateless nature of the recursion. However, prior work realizes this insight only partially. For instance, [BHKZ25] provides a construction restricted to R1CS, limiting its expressiveness compared to generalized constraint systems such as CCS. Furthermore, it is formulated exclusively for IVC (accumulating single pairs) rather than the broader PCD setting, is bound to multilinear encodings, and incurs the overhead of committing to cross-term polynomials. These limitations leave open the challenge of constructing a modular, stateless accumulation framework that supports general constraint systems.

1.1 Our Contributions

Holography Accumulation. We introduce *holography accumulation*, a framework that accumulates the holographic checks performed by lincheck-based SNARK verifiers while keeping the recursive prover stateless,

thereby enabling stateless recursion without requiring a full SNARK proof at each step. Our construction builds on the approach of [BHKZ25], where the authors construct separate accumulation schemes for the witness and public structure statements, and generalizes it in several directions:

1. **Broad Arithmetization Support:** We extend holographic accumulation beyond R1CS to Customizable Constraint Systems (CCS), capturing a wide class of modern SNARKs.
2. **Proof-Carrying Data:** We present our accumulation scheme for the multi-instance case (PCD) instead of the two-instance one (IVC).
3. **Flexible Instantiation:** Our scheme supports both multilinear and univariate representations, allowing the construction to be optimized for the underlying polynomial commitment scheme.
4. **Commitment-Free Holography Accumulation:** We present a scheme where the accumulation of the holographic checks does not require auxiliary commitments. This reduces the prover's cryptographic overhead and simplifies the protocol.

By decoupling these checks from the witness, we eliminate the dependence on the circuit structure in the recursive step while preserving the privacy required for distributed computation.

Generalized Bilinear Forms. We abstract the verification procedures of many SNARK constructions based on the lincheck argument as checking membership in a linear-algebraic relation $\mathcal{R}_{\mathsf{GBF}}$, providing a unifying representation of their holographic verification procedures. We provide two concrete interactive protocols Π_{GBF1} and Π_{GBF2}, which cover the different variable representations mentioned above. We show how to instantiate these protocols to recover and generalize SuperMarlin and SuperSpartan [STW23], as well as accumulation schemes for the holographic checks, providing uniform representation and security analysis of our protocols. As a byproduct, we also describe how to adapt Π_{GBF2} to recover early-stopping schemes such as the early stopping variant of SuperSpartan used in the Hypernova folding scheme [KS24].

Generic PCD Construction. Using the above ingredients and compilers from [BCMS20, Thm. 5.2] and [BMNW25, Thm. 4.3], we construct PCD schemes that work with any polynomial commitment scheme (univariate or multivariate) that supports evaluation proof accumulation (e.g. KZG [KZG10], Pedersen [BCMS20], and their variants) or whose evaluation proof verification can be efficiently arithmetized (e.g. FRI [BBHR18] and its variants, STIR [ACFY24], WHIR [ACFY25], Binius [DP25], and Basefold [ZCF24]). Unlike folding-based IVC and PCD, our construction allows a prover to generate a proof for step i using only the proofs of step $i-1$ and public accumulators, without knowing the previous step's witnesses. This makes our scheme suitable for privacy-preserving applications where provers do not share state.

Efficient Decider. Finally, we introduce a novel "Decider" strategy to terminate the recursion in the presence of a homomorphic commitment scheme for the matrix polynomials. Instead of running the SNARK and accumulation verifiers, our decider runs a lightweight interaction with the prover that reduces the accumulated holographic checks to a single polynomial evaluation check. The cryptographic cost of the interaction for the decider is dominated by a linear combination of the matrix commitments. This makes our scheme particularly advantageous for non-uniform IVC/PCD, where the matrices corresponding to different functions need to be evaluated at different points. Concretely, for ℓ functions each described by t CCS matrices, our decider performs a single linear combination of the $\ell \cdot t$ commitments and checks a single evaluation proof, instead of computing $2\ell \cdot t$ matrix evaluations.

1.2 Further Comparison

Folding schemes constitute the state-of-the-art constructions for *stateful* recursion. This stems from the fact that both the statement prover and the accumulation prover and verifier are very efficient. Nevertheless,

they come with significant shortcomings due to the need to communicate the long, private state between the provers. On the other hand, full recursion and atomic accumulation suffer from more expensive statement provers and higher recursive overhead due to the need to produce and prove/accumulate in circuit a full SNARK proof. Holography accumulation aims to bridge the gap between the two approaches by having the prover produce only a *proof-succinct* NARK. Concretely, for matrices of sparsity K and witness size n, when this methodology is applied to the full recursion scheme of Fractal [COS20] or the recursive construction of Marlin [CHM+20] using KZG [KZG10] commitments and the corresponding accumulation scheme of [BCMS20], the prover does not need to commit and prove evaluations of the structure related polynomials, which have degree $6K - 6$. This directly translates to $\mathcal{O}(n)$ cryptographic asymptotic costs for the prover instead of $\mathcal{O}(K)$. Moreover, the recursion remains stateless, and the accumulator size grows by a constant amount in the univariate case and logarithmically in the multivariate one.

Further comparison with [BHKZ25]. As stated in the introduction, the accumulation scheme for evaluations of the computation polynomials presented in [BHKZ25] is limited to the R1CS relation, IVC setting (accumulation of a single pair of values), and multilinear encoding. In more detail, let $\nu = \log n$, $M(\mathbf{X}, \mathbf{Y})$ be a 2ν-variate multilinear polynomial encoding a matrix $\mathbf{M} \in \mathbb{F}^{n \times n}$ and $\mathbf{r}_1 = (\mathbf{r}_{x1}, \mathbf{r}_{y1}), \mathbf{r}_2 = (\mathbf{r}_{x2}, \mathbf{r}_{y2}) \in \mathbb{F}^{2\nu}$ be two evaluation points. The authors of [BHKZ25] accumulate the two claimed evaluations $M(\mathbf{r}_1) = u_1$ and $M(\mathbf{r}_2) = u_2$ to a single evaluation claim $M(\mathbf{r}) = u$, where $\mathbf{r}$ is a *structured* random linear combination of $\mathbf{r}_1$ and $\mathbf{r}_2$. In order for the verifier to compute the resulting claim, the prover must provide an *error term* polynomial of degree $2\nu - 2$ which the verifier has to evaluate. In [BHKZ25] this process is performed for all three of the R1CS matrices, leading to $6 \cdot \nu - 6$ field elements. We note that if a single verifier randomness is used for all the matrices then the resulting evaluation point will be the same for all of them. Their 2-folding scheme can be extended to a K-folding scheme required to achieve PCD for general graphs with total communication cost $6 \cdot K \cdot (\nu - 1)$, either by assuming a $\log K$-depth accumulation tree or by extending their structured linear combination to account for K terms.

In contrast, our schemes leverage the sumcheck protocol in order to reduce multiple evaluation claims to a single one about a *fresh* evaluation point. Because a sumcheck protocol exists both for multivariate and univariate polynomials, we can instantiate our holography accumulation with both types of commitment schemes, obtaining different efficiency trade-offs. Moreover, because of the batching properties of the sumcheck protocol, we avoid the multiplicative dependency between the number of accumulated instances K and the size of the witness. More specifically, one of our constructions that uses multivariate polynomials has communication cost for R1CS $7 \cdot \nu + 3 \cdot (K + 1)$ field elements and no cryptographic costs.

2 Techniques

Preliminaries and Notation. *Algebraic Structures.* For $n \in \mathbb{N}$ and finite field $\mathbb{F}$, we define $\mathbb{H}$ as the evaluation domain. We use the parameter $\nu \in \{1, \log n\}$ to distinguish between protocols for univariate and $\log n$-variate polynomials, respectively. In the univariate case ($\nu = 1$), $\mathbb{H}$ is the multiplicative subgroup of n-th roots of unity in $\mathbb{F}$, and the boolean hypercube $\mathcal{B}_{\log n} := \{0,1\}^{\log n}$ otherwise. We also assume a canonical ordering of $\mathbb{H}$. We use boldface for elements $\mathbf{h} \in \mathbb{H}$, elements $\boldsymbol{\alpha} \in \mathbb{F}^\nu$, and variables $\mathbf{X}, \mathbf{Y}$ for both the univariate and multivariate cases. We omit boldface only where it's explicit that we work in the univariate case.

We use boldface to denote vectors $\mathbf{u}$ and matrices $\mathbf{M}$. We use $\mathbf{u}^\top \mathbf{v}$ to denote the inner-product of $\mathbf{u}$ and $\mathbf{v}$, $\mathbf{u} \circ \mathbf{v}$ to denote their Hadamard product and $\bigcirc_{i=1}^{n} \mathbf{u}_i = \mathbf{u}_1 \circ \ldots \circ \mathbf{u}_n$. We denote the vanishing polynomial in $\mathbb{H}$ as $u_{\mathbb{H}}(\mathbf{X})$ and the Lagrange basis polynomial for $\mathbf{h} \in \mathbb{H}$ as $\lambda_{\mathbf{h}}(\mathbf{X})$. Specifically, $\lambda_{\mathbf{h}}(\mathbf{X})$ refers to the univariate polynomial $\frac{h}{n} \frac{u_{\mathbb{H}}(X)}{X - h}$ when $\nu = 1$, and the equality polynomial $\prod_{i=0}^{\nu-1}(h_i X_i + (1 - h_i)(1 - X_i))$ when $\nu = \log n$.

We define the polynomial associated with a vector $\mathbf{z} \in \mathbb{F}^n$ as $z(\mathbf{X}) = \mathbf{z}^\top \boldsymbol{\lambda}(\mathbf{X}) \in \mathbb{F}^{\leq d}[\mathbf{X}]$, where the maximum degree d equals $n - 1$ when $\nu = 1$, and $d = 1$ (in each variable) when $\nu = \log n$. Similarly, the

polynomial associated with a matrix $\mathbf{M} \in \mathbb{F}^{n \times n}$ is $M(\mathbf{X}, \mathbf{Y}) = \boldsymbol{\lambda}(\mathbf{Y})^\top \mathbf{M} \boldsymbol{\lambda}(\mathbf{X})$. A key property we will use repeatedly is that the identity matrix polynomial $\Lambda(\mathbf{X}, \mathbf{Y}) := I(\mathbf{X}, \mathbf{Y})$ has a compact representation in both cases –namely $\frac{u_{\mathbb{H}}(X)Y - u_{\mathbb{H}}(Y)X}{n(X-Y)}$ [CFF+21] and $\prod_{i=0}^{\nu-1}(X_i Y_i + (1-X_i)(1-Y_i))$, respectively– and thus can be evaluated in $\mathcal{O}(\log n)$ time. We assume polynomial commitment schemes PC and PC_2 for ν-variate and 2ν-variate polynomials, respectively. We write $[c]$ for a commitment to c.

Indexed relations. A parametrized, indexed relation $\mathcal{R}(\mathsf{pp})$ is a set of triples $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}(\mathsf{pp})$, where pp are public parameters, $\mathbb{i}$ is the index, $\mathbb{x}$ is the instance, and $\mathbb{w}$ is the witness. The multi-instance relation $\mathcal{R}^*$ is defined to be the set of tuples $(\mathbb{i}, (\mathbb{x}^{(1)}, \ldots, \mathbb{x}^{(m)}), (\mathbb{w}^{(1)}, \ldots, \mathbb{w}^{(m)}))$ such that $\forall i \in [m], (\mathbb{i}, \mathbb{x}^{(i)}, \mathbb{w}^{(i)}) \in \mathcal{R}$. The **truth relation** is defined to be $\mathcal{R}_\top := (\bot, \mathsf{true}, \bot)$. For a relation $\mathcal{R}$ we define the corresponding language $\mathcal{L}(\mathcal{R}) = \{(\mathbb{i}, \mathbb{x}) | \exists \mathbb{w}, (\mathbb{i}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}\}$.

Reductions of knowledge. A (public-coin, indexed) interactive reduction of knowledge $\Pi : \mathcal{R} \to \mathcal{R}'$ [KP23] is an interactive protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ which reduces the claim that $(\mathbb{i}, \mathbb{x}) \in \mathcal{L}(\mathcal{R})$ to a new claim $(\mathbb{i}', \mathbb{x}') \in \mathcal{L}(\mathcal{R}')$. Moreover, if $\mathcal{P}$ knows a witness $\mathbb{w}'$ such that $(\mathbb{i}', \mathbb{x}', \mathbb{w}') \in \mathcal{R}'$, then it must also know a witness $\mathbb{w}$ such that $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}$. As noted in [KP23], reductions of knowledge

1. compose sequentially: if $\Pi_1 : \mathcal{R} \to \mathcal{R}'$, $\Pi_2 : \mathcal{R}' \to \mathcal{R}''$ are reductions of knowledge, then $\Pi_2 \circ \Pi_1 : \mathcal{R} \to \mathcal{R}''$ is a reduction of knowledge, where $\Pi_2 \circ \Pi_1$ denotes first using Π_1 to reduce $\mathbb{x} \in \mathcal{L}(\mathcal{R})$ to some $\mathbb{x}' \in \mathcal{L}(\mathcal{R}')$ and then using Π_2 to reduce the latter to some $\mathbb{x}'' \in \mathcal{L}(\mathcal{R}'')$,
2. compose in parallel: if $\Pi_1 : \mathcal{R} \to \mathcal{R}'$, $\Pi_2 : \mathcal{R}'' \to \mathcal{R}'''$ are reductions of knowledge, then $\Pi_1 \times \Pi_2 : \mathcal{R} \times \mathcal{R}'' \to \mathcal{R}' \times \mathcal{R}'''$ is a reduction of knowledge, where $\Pi_1 \times \Pi_2$ denotes executing in parallel Π_1, Π_2 and $\mathcal{R}_1 \times \mathcal{R}_2$ is the relation:

$$\mathcal{R} = \{((\mathbb{i}_1, \mathbb{i}_2), (\mathbb{x}_1, \mathbb{x}_2), (\mathbb{w}_1, \mathbb{w}_2)) \mid (\mathbb{i}_1, \mathbb{x}_1, \mathbb{w}_1) \in \mathcal{R}_1 \text{ and } (\mathbb{i}_2, \mathbb{x}_2, \mathbb{w}_2) \in \mathcal{R}_2\}$$

Moreover, we define the identity reduction of knowledge $\mathsf{ID}_{\mathcal{R}} : \mathcal{R} \to \mathcal{R}$ where $\mathcal{P}$ and $\mathcal{V}$ interact trivially exchanging no messages and outputting $\mathbb{x}$ and $\mathbb{w}$. Finally, we call a reduction $\Pi : \mathcal{R} \to \mathcal{R}_\top$ an argument of knowledge.

2.1 Decoupling Structure and Witness

To achieve stateless accumulation, we must identify which components of the verification logic depend on the witness and which depend only on public parameters. We focus our analysis on two prominent SNARK families: Marlin [CHM+20] and Spartan [Set20]. We choose these as representative examples because they arrive at the same requirement, i.e. correctness of witness and matrix polynomial evaluations, but through different routes. In both systems, the verifier checks that a witness vector $\mathbf{z}$ satisfies a system of equations defined by circuit matrices. Standard folding schemes fold the pair (Matrix, Witness) together. Our goal is to inspect the algebraic structure of these checks to separate the public matrix operations from the private witness vector.

Both approaches rely heavily on the sumcheck argument [LFKN92, BCR+19]. This is an interactive protocol that reduces the claim $\sum_{\mathbf{h} \in \mathbb{H}} g(\mathbf{h}) = s$ to that of the correctness of an evaluation $g(\boldsymbol{\alpha})$ at a random point chosen by $\mathcal{V}$. The sumcheck argument is known to exist for both the univariate and the multivariate case, and it can be used to prove inner-product relations between vectors, since $\mathbf{u}^\top \mathbf{v} = \sum_{\mathbf{h} \in \mathbb{H}} \mathbf{u}^\top \boldsymbol{\lambda}(\mathbf{h}) \cdot \mathbf{v}^\top \boldsymbol{\lambda}(\mathbf{h}) = \sum_{\mathbf{h} \in \mathbb{H}} g(\mathbf{h})$, where $g(\mathbf{X}) := \mathbf{u}^\top \boldsymbol{\lambda}(\mathbf{X}) \cdot \mathbf{v}^\top \boldsymbol{\lambda}(\mathbf{X})$. Moreover, sumcheck arguments for different polynomials can be batched in a single one with negligible soundness error. An important fact that was originally exploited in [KS24] is that, by batching the sumcheck arguments, the evaluations of the polynomials that $\mathcal{V}$ needs to check are for the same evaluation point. Looking ahead, our schemes for accumulating the holographic checks rely on this fact. In all the protocols we implicitly assume that the sumchecks are batched. We avoid making it explicit for ease of exposition.

Warm-up: Marlin and Spartan for R1CS. A proof of knowledge for the R1CS relation requires $\mathcal{P}$ to prove knowledge of a vector $\mathbf{z} \in \mathbb{F}^n$ such that $(\mathbf{Az}) \circ (\mathbf{Bz}) = \mathbf{Cz}$, where $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{F}^{n \times n}$ are the R1CS matrices encoding the circuit that captures the desired computation (we omit the public input part of $\mathbf{z}$ in this overview). Both approaches share the same starting point: $\mathcal{P}$ computes and sends to $\mathcal{V}$ a polynomial commitment $[z(\mathbf{X})]$ to $z(\mathbf{X}) = \mathbf{z}^\top \boldsymbol{\lambda}(\mathbf{X})$. After this step the two protocols deviate. We highlight how.

The Marlin approach. This approach is based on the following fact: proving that $\mathbf{z}$ is an R1CS witness is equivalent to proving that, for $M \in \{A, B, C\}$, the vectors $\mathbf{z}_M$ (to which $\mathcal{P}$ has provided commitments via $\mathbf{z}_M^\top \boldsymbol{\lambda}(\mathbf{X})$) satisfy

$$\text{(a) } \mathbf{z}_A \circ \mathbf{z}_B - \mathbf{z}_C = \mathbf{0} \qquad \text{(b) } \mathbf{z}_M = \mathbf{Mz}, \quad M \in \{A, B, C\} \tag{1}$$

Furthermore, it is known [BCR+19] that vector equality is implied (with overwhelming probability) from the equality of their inner-products with a sufficiently random vector $\mathbf{r} \in \mathbb{F}^n$. The sparse vector $\mathbf{r} = \boldsymbol{\lambda}(\boldsymbol{\alpha})$ is used. Hence, given verifier randomness, equations (1.a) and (1.b) are reduced to proving that

$$\text{(a) } \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top (\mathbf{z}_A \circ \mathbf{z}_B - \mathbf{z}_C) = 0 \qquad \text{(b) } \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{z}_M = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{Mz} \tag{2}$$

Now, using the equalities $\mathbf{u}^\top \mathbf{v} = \sum_{\mathbf{h} \in \mathbb{H}} \mathbf{u}^\top \boldsymbol{\lambda}(\mathbf{h}) \cdot \mathbf{v}^\top \boldsymbol{\lambda}(\mathbf{h})$ and $\boldsymbol{\lambda}(\mathbf{h})^\top (\mathbf{u} \circ \mathbf{v}) = \boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{u} \cdot \boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{v}$, and the definition of $\Lambda(\mathbf{X}, \mathbf{Y}) = \boldsymbol{\lambda}(\mathbf{Y})^\top \boldsymbol{\lambda}(\mathbf{X})$, equations (2.a) and (2.b) are in turn reduced to proving that

$$\sum_{\mathbf{h} \in \mathbb{H}} \Lambda(\boldsymbol{\alpha}, \mathbf{h}) \cdot (\boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{z}_A \cdot \boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{z}_B - \boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{z}_C) = 0 \tag{3}$$

$$\sum_{\mathbf{h} \in \mathbb{H}} \Lambda(\boldsymbol{\alpha}, \mathbf{h}) \cdot \boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{z}_M - \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M} \boldsymbol{\lambda}(\mathbf{h}) \cdot \mathbf{z}^\top \boldsymbol{\lambda}(\mathbf{h}) = 0 \tag{4}$$

Define the polynomials $g_{\mathsf{IP}}(\mathbf{X}) \coloneqq \Lambda(\boldsymbol{\alpha}, \mathbf{X}) \cdot (\boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{z}_A \cdot \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{z}_B - \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{z}_C)$ and $g_{\mathsf{LinM}}(\mathbf{X}) \coloneqq \Lambda(\boldsymbol{\alpha}, \mathbf{X}) \cdot \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{z}_M - \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M} \boldsymbol{\lambda}(\mathbf{X}) \cdot \mathbf{z}^\top \boldsymbol{\lambda}(\mathbf{X})$. Then, equations 3 and 4 can be proven via a sumcheck argument on these polynomials. At the end of the sumcheck, $\mathcal{P}$ assists $\mathcal{V}$ with computing evaluations $g_{\mathsf{IP}}(\boldsymbol{\beta})$ and $g_{\mathsf{LinM}}(\boldsymbol{\beta})$ by providing $z_\beta, z_{\beta,M}$, and v_M, purported to be $z(\boldsymbol{\beta})$, $z_M(\boldsymbol{\beta})$, and $\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M} \boldsymbol{\lambda}(\boldsymbol{\beta})$, respectively, which $\mathcal{V}$ now needs to verify. For the correctness of v_M, $\mathcal{V}$ can compare it with $\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M} \boldsymbol{\lambda}(\boldsymbol{\beta})$ (leading to a proof succinct but not verification succinct argument of knowledge) or by engaging in a holographic argument with $\mathcal{P}$.

The Spartan approach. This approach avoids the use of the intermediate vectors $\mathbf{z}_M$ at the cost of an extra sumcheck invocation. In more detail, $\mathcal{V}$ provides randomness $\boldsymbol{\alpha} \in \mathbb{F}^\nu$ and reduces the R1CS equation to proving that

$$\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top ((\mathbf{Az}) \circ (\mathbf{Bz}) - \mathbf{Cz}) = 0 \tag{5}$$

which is equivalent to proving that

$$\sum_{\mathbf{h} \in \mathbb{H}} \Lambda(\boldsymbol{\alpha}, \mathbf{h}) \cdot \boldsymbol{\lambda}(\mathbf{h})^\top ((\mathbf{Az}) \circ (\mathbf{Bz}) - \mathbf{Cz}) = 0 \iff$$

$$\sum_{\mathbf{h} \in \mathbb{H}} \Lambda(\boldsymbol{\alpha}, \mathbf{h}) \cdot \left(\boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{Az} \cdot \boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{Bz} - \boldsymbol{\lambda}(\mathbf{h})^\top \mathbf{Cz} \right) = 0 \iff \sum_{\mathbf{h} \in \mathbb{H}} g(\mathbf{h}) = 0$$

where $g(\mathbf{X}) \coloneqq \Lambda(\boldsymbol{\alpha}, \mathbf{X}) \cdot \left(\boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{Az} \cdot \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{Bz} - \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{Cz} \right)$. Hence, $\mathcal{P}$ and $\mathcal{V}$ can run the sumcheck argument to reduce the statement to the correctness of evaluation $g(\boldsymbol{\beta})$. $\mathcal{P}$ assists $\mathcal{V}$ in computing said

evaluation by providing d_M purported to be $\boldsymbol{\lambda}(\boldsymbol{\beta})^\top \mathbf{M}\mathbf{z}$. $\mathcal{V}$ now needs to verify its correctness, i.e. that d_M is indeed equal to $\boldsymbol{\lambda}(\boldsymbol{\beta})^\top \mathbf{M}\mathbf{z}$. Observe that

$$d_M = \boldsymbol{\lambda}(\boldsymbol{\beta})^\top \mathbf{M}\mathbf{z} \iff d_M = \sum_{\mathbf{h}\in\mathbb{H}} \boldsymbol{\lambda}(\boldsymbol{\beta})^\top \mathbf{M}\boldsymbol{\lambda}(\mathbf{h}) \cdot \mathbf{z}^\top \boldsymbol{\lambda}(\mathbf{h}) \iff \sum_{\mathbf{h}\in\mathbb{H}} g'_M(\mathbf{h}) = d_M$$

where $g'_M(\mathbf{X}) := \boldsymbol{\lambda}(\boldsymbol{\beta})^\top \mathbf{M}\boldsymbol{\lambda}(\mathbf{X}) \cdot \mathbf{z}^\top \boldsymbol{\lambda}(\mathbf{X})$. Therefore, $\mathcal{P}$ and $\mathcal{V}$ can run another sumcheck argument to prove the correctness of d_M, at the end of which $\mathcal{P}$ provides values z_β and v_M purported to be $z(\boldsymbol{\gamma})$ and $\boldsymbol{\lambda}(\boldsymbol{\beta})^\top \mathbf{M}\boldsymbol{\lambda}(\boldsymbol{\gamma})$, respectively, which $\mathcal{V}$ now needs to verify. For the correctness of v_M, it either computes and compares $\boldsymbol{\lambda}(\boldsymbol{\beta})^\top \mathbf{M}\boldsymbol{\lambda}(\boldsymbol{\gamma})$ to it (leading to a proof succinct but not verification succinct argument of knowledge) or can engage in a holographic argument with $\mathcal{P}$.

The above analysis extends in a straightforward manner to the SuperMarlin and SuperSpartan [STW23] variants of the protocols for proving that the CCS relation holds for $\mathbf{z}$, namely that

$$\sum_{i=1}^{q} c_i \bigcirc_{j\in S_i} \mathbf{M}_j \mathbf{z} = \mathbf{0} \tag{6}$$

Generalization of CCS and Holographic Checks. The preceding exposition shows that both approaches essentially reduce the R1CS/CCS relation to intermediate "linear algebra equations" ((2.a), (2.b) and (5)) which in turn are reduced to the correctness of two sets of evaluations: (1) witness dependent committed polynomials, and (2) public (holographic) matrix polynomials. An important observation is that the latter is itself a similar "linear algebra equation", namely $\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}\boldsymbol{\lambda}(\boldsymbol{\beta}) = m$. Looking ahead, our goal is to define a relation that captures both types of equations and design a reduction from it that can be used both for circuit satisfiability and for accumulating the holographic checks.

Informally, given commitment key ck of a ν-variate polynomial commitment scheme PC, we define the *polynomial commitment evaluation relation* $\mathcal{R}_{\mathsf{PCE}}(\mathsf{ck})$ that has an empty index, instance that includes (prover generated) commitment $[p(\mathbf{X})]$, evaluation point $\boldsymbol{\alpha} \in \mathbb{F}^\nu$, and claimed evaluation $\gamma \in \mathbb{F}$, and witness being the ν-variate polynomial $p(\mathbf{X})$. Moreover, given commitment key ck_2 of a 2ν-variate polynomial commitment scheme PC_2, we define the *holographic bivariate polynomial commitment evaluation relation* $\mathcal{R}_{\mathsf{hbPCE}}(\mathsf{ck}_2)$ which has an index consisting of a set of size t_M of 2ν-variate polynomials $P_i(\mathbf{X}, \mathbf{Y})$, instance that includes (honestly generated) commitments $[P_i(\mathbf{X}, \mathbf{Y})]$, claimed evaluations γ_i, an evaluation point $(\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{F}^{2\nu}$, and an empty witness. Then, both SuperMarlin and SuperSpartan can be viewed as interactive reductions of knowledge from $\mathcal{R}_{\mathsf{CCS}}$ to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$.

Now, assume a batching randomness $\boldsymbol{\eta} \in \mathbb{F}^{t_M}$ and consider the linear combination

$$\sum_{i\in[t_M]} \eta_i M_i(\boldsymbol{\beta}, \boldsymbol{\alpha}) = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \left(\sum_{i\in[t_M]} \eta_i \mathbf{M}_i \boldsymbol{\lambda}(\boldsymbol{\beta}) \right) = \sum_{i\in[t_M]} \eta_i \gamma_i := \gamma \tag{7}$$

and observe that it is reminiscent of what $\mathcal{P}$ needs to prove in SuperMarlin and SuperSpartan after the initial randomness $\boldsymbol{\alpha} \in \mathbb{F}^\nu$ from $\mathcal{V}$, namely that

$$\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \left(\sum_{i=1}^{q} c_i \bigcirc_{j\in S_i} \mathbf{M}_j \mathbf{z} \right) = 0 \tag{8}$$

Indeed, one can apply the rest of the protocols' workflow to the statement of equation 7 and reduce it to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$ as well. This may appear pointless at first glance, but at the end of the protocol the statement of $\mathcal{R}_{\mathsf{hbPCE}}$ is about claimed evaluations $\gamma'_i \in \mathbb{F}$ of the matrices *at a fresh point* $(\boldsymbol{\alpha}', \boldsymbol{\beta}') \in \mathbb{F}^{2\nu}$. Furthermore, since the protocols only run sumchecks, their invocations on claims of the form of equation 7 can be batched. As a result, we can reduce multiple claims of the form $\sum_{i\in[t_M]} \eta_i M_i(\boldsymbol{\beta}^{(j)}, \boldsymbol{\alpha}^{(j)}) = \gamma^{(j)}$ to

claims $M_i(\boldsymbol{\beta}', \boldsymbol{\alpha}') = \gamma_i'$ and then back to a claim $\sum_{i\in[t_M]} \eta_i' M_i(\boldsymbol{\beta}', \boldsymbol{\alpha}') = \gamma' =: \sum_{i\in[t_M]} \eta_i' \gamma_i'$. In a sense, we can use the subroutines in SuperMarlin and SuperSpartan to *reduce multiple holographic statements (i.e. matrix evaluation claims for multiple points) to a single, fresh holographic statement*. Looking ahead, this will allow us to *efficiently accumulate* such statements.

Motivated by the above, we isolate the subprotocols that achieve this and we present them as interactive reductions of knowledge from a linear algebra relation $\mathcal{R}_{\mathsf{GBF}}$ to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$, where $\mathcal{R}_{\mathsf{GBF}}$ simultaneously generalizes $\mathcal{R}_{\mathsf{CCS}}$ and $\mathcal{R}_{\mathsf{hbPCE}}$ (in the sense that both can be reduced to the former). In fact, we construct said reductions starting from the multi-instance relation $\mathcal{R}^*_{\mathsf{GBF}}$ and show how this can be used to accumulate multiple holographic statements into a new one. Moreover, we construct a reduction from $\mathcal{R}_{\mathsf{CCS}}$ to $\mathcal{R}_{\mathsf{GBF}}$. Looking ahead, the $\mathcal{R}_{\mathsf{GBF}}$ relation and its reductions provide us with enough flexibility to modularly construct and analyze the security of complex schemes such as IVC [Val08] and PCD [CT10].

2.2 Generalized Bilinear Forms

We define the (parametrized, indexed) generalized bilinear forms relation which captures the inner-product relation between a sum of Hadamard products and a sum of Hadamard products of column space vectors. This is a flexible enough linear algebra relation to capture the intermediate equations when proving $\mathcal{R}_{\mathsf{CCS}}$ as well as the holographic checks. In more detail, the relation $\mathcal{R}_{\mathsf{GBF}}$ is parametrized by the commitment keys of a ν-variate and a 2ν-variate polynomial commitment scheme PC and PC_2, respectively. Its index is a set of matrices $\mathcal{M}$. Its instance includes a value $s \in \mathbb{F}$, the structure of the inner-product vectors, and commitments to the component vectors, as well as (honestly generated) commitments to the matrix polynomials. Finally, the witness consists of the component vectors, such that

$$s = \left(\sum_{i=1}^{q_l} c_{l,i} \bigcirc_{j\in S_{l,i}} \mathbf{u}_j\right)^\top \left(\sum_{i=1}^{q_r} c_{r,i} \bigcirc_{(j_M,j_v)\in S_{r,i}} \mathbf{M}_{j_M} \mathbf{v}_{j_v}\right). \tag{9}$$

We also consider special cases where the vectors $\mathbf{u}, \mathbf{v}$ correspond to known polynomials that the verifier can evaluate on its own without the help of the prover. In such cases, sharing the commitments and evaluation proofs is unnecessary. Concretely, we consider the cases where either or both $\mathbf{u}$ and $\mathbf{v}$ correspond to $\boldsymbol{\lambda}(\boldsymbol{\alpha})$ for some $\boldsymbol{\alpha} \in \mathbb{F}^\nu$. The corresponding special cases of $\mathcal{R}_{\mathsf{GBF}}$ are $\mathcal{R}_{\mathsf{GBF},\alpha}$ and $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ defined by the following equations, respectively:

$$\text{(a) } s = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \left(\sum_{i=1}^{q_r} c_{r,i} \bigcirc_{(j_M,j_v)\in S_{r,i}} \mathbf{M}_{j_M} \mathbf{v}_{j_v}\right) \text{(b) } s = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \left(\sum_{i\in[t_M]} c_i \mathbf{M}_i \boldsymbol{\lambda}(\boldsymbol{\beta})\right) \tag{10}$$

In both $\mathcal{R}_{\mathsf{GBF},\alpha}$ and $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ we replace the commitments to $\boldsymbol{\lambda}(\boldsymbol{\alpha})$ and $\boldsymbol{\lambda}(\boldsymbol{\beta})$ with $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, as they serve as implicit commitments to them. We call the linear combination reduction of equation 7 from $\mathcal{R}_{\mathsf{hbPCE}}$ to $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ Π_{batchM}.

We construct a family of interactive reductions of knowledge Π_{GBF}. This consists of two interactive reductions of knowledge, Π_{GBF1} and Π_{GBF2}, from $\mathcal{R}_{\mathsf{GBF}}$ to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$, which work for both univariate and multivariate polynomials. We now give a high level overview of the two protocols. In what follows, S_R denotes the set that contains the pairs (j_M, j_v) such that $\exists i \in [q_r]$ with $(j_M, j_v) \in S_{r,i}$ and $t_{M,v} = |S_R|$.

Π_{GBF1}. As in the Marlin approach for R1CS, $\mathcal{P}$ sends polynomial commitments to the intermediate vectors $\mathbf{d}_{j_M,j_v} = \mathbf{M}_{j_M} \mathbf{v}_{j_v}$, for $(j_M, j_v) \in S_R$, reducing $\mathcal{R}_{\mathsf{GBF}}$ to the following:

$$s = \left(\sum_{i=1}^{q_l} c_{l,i} \bigcirc_{j\in S_{l,i}} \mathbf{u}_j\right)^\top \left(\sum_{i=1}^{q_r} c_{r,i} \bigcirc_{(j_M,j_v)\in S_{r,i}} \mathbf{d}_{j_M,j_v}\right) \tag{11}$$

$$\mathbf{d}_{j_M,j_v} = \mathbf{M}_{j_M}\mathbf{v}_{j_v}, \quad \forall (j_M, j_v) \in S_R \tag{12}$$

Observe that the right-hand side of equation 11 is equal to

$$\sum_{\mathbf{h}\in\mathbb{H}} \left(\sum_{i=1}^{q_l} c_{l,i} \bigcirc_{j\in S_{l,i}} \mathbf{u}_j\right)^\top \boldsymbol{\lambda}(\mathbf{h}) \cdot \left(\sum_{i=1}^{q_r} c_{r,i} \bigcirc_{(j_M,j_v)\in S_{r,i}} \mathbf{d}_{j_M,j_v}\right)^\top \boldsymbol{\lambda}(\mathbf{h}) \tag{13}$$

which in turn is equal to

$$\sum_{\mathbf{h}\in\mathbb{H}} \left(\sum_{i=1}^{q_l} c_{l,i} \prod_{j\in S_{l,i}} \mathbf{u}_j^\top \boldsymbol{\lambda}(\mathbf{h})\right) \cdot \left(\sum_{i=1}^{q_r} c_{r,i} \bigcirc_{(j_M,j_v)\in S_{r,i}} \mathbf{d}_{j_M,j_v}^\top \boldsymbol{\lambda}(\mathbf{h})\right). \tag{14}$$

Hence, equation 11 can be proven via a sumcheck argument on the polynomial

$$g_{\mathsf{IP}}(\mathbf{X}) \coloneqq \left(\sum_{i=1}^{q_l} c_{l,i} \prod_{j\in S_{l,i}} \mathbf{u}_j^\top \boldsymbol{\lambda}(\mathbf{X})\right) \cdot \left(\sum_{i=1}^{q_r} c_{r,i} \prod_{(j_M,j_v)\in S_{r,i}} \mathbf{d}_{j_M,j_v}^\top \boldsymbol{\lambda}(\mathbf{X})\right).$$

Moreover, given verifier randomness $\boldsymbol{\alpha} \in \mathbb{F}^\nu$, 12 holds (with overwhelming probability) as long as

$$\mathbf{d}_{j_M,j_v}^\top \boldsymbol{\lambda}(\boldsymbol{\alpha}) = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M}\mathbf{v}_{j_v} \iff \sum_{\mathbf{h}\in\mathbb{H}} g_{\mathsf{Lin}}(\mathbf{h}) = 0 \tag{15}$$

where $g_{\mathsf{Lin}}(\mathbf{X}) \coloneqq \mathbf{d}_{j_M,j_v}^\top \boldsymbol{\lambda}(\mathbf{X}) \cdot \Lambda(\boldsymbol{\alpha},\mathbf{X}) - \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M}\boldsymbol{\lambda}(\mathbf{X}) \cdot \mathbf{v}_{j_v}^\top \boldsymbol{\lambda}(\mathbf{X})$. So, $\mathcal{P}$ and $\mathcal{V}$ can engage in a sumcheck argument, reducing the statement to the correctness of $g_{\mathsf{IP}}(\boldsymbol{\beta})$ and $g_{\mathsf{Lin}}(\boldsymbol{\beta})$. $\mathcal{P}$ assists $\mathcal{V}$ in computing these by providing values u_i, v_i, d_{j_M,j_v}, and m_i purported to be $u_i(\boldsymbol{\beta})$, $v_i(\boldsymbol{\beta})$, $d_{j_M,j_v}(\boldsymbol{\beta})$, and $\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_i \boldsymbol{\lambda}(\boldsymbol{\beta})$, respectively, and $\mathcal{V}$ is left with checking their correctness.

Π_{GBF2}. Here we follow the Spartan approach for R1CS after the initial randomness, namely $\mathcal{P}$ and $\mathcal{V}$ engage in a sumcheck argument for the claim $\sum_{\mathbf{h}\in\mathbb{H}} g(\mathbf{h}) = 0$, where

$$g(\mathbf{X}) \coloneqq \left(\sum_{i=1}^{q_l} c_{l,i} \prod_{j\in S_{l,i}} \mathbf{u}_j^\top \boldsymbol{\lambda}(\mathbf{X})\right) \cdot \left(\sum_{i=1}^{q_r} c_{r,i} \prod_{(j_M,j_v)\in S_{r,i}} \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{M}_{j_M}\mathbf{v}_{j_v}\right).$$

In the end, $\mathcal{P}$ assists $\mathcal{V}$ with computing $g(\boldsymbol{\alpha})$ by providing values u_i and d_{j_M,j_v} purported to be $u_i(\boldsymbol{\alpha})$ and $\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M}\mathbf{v}_{j_v}$, respectively. $\mathcal{V}$ now has to check their correctness. To do so, observe that

$$d_{j_M,j_v} = \sum_{\mathbf{h}\in\mathbb{H}} \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M}\boldsymbol{\lambda}(\mathbf{h}) \cdot \mathbf{v}_{j_v}^\top \boldsymbol{\lambda}(\mathbf{h}) = \sum_{\mathbf{h}\in\mathbb{H}} g_{j_M,j_v}(\mathbf{h}). \tag{16}$$

Hence, $\mathcal{P}$ and $\mathcal{V}$ can run another sumcheck argument for the polynomial $g_{j_M,j_v}(\mathbf{X})$ to reduce the statement to the correctness of $g_{j_M,j_v}(\boldsymbol{\beta})$. $\mathcal{P}$ assists $\mathcal{V}$ in computing the evaluation by providing values v_{j_v}, and m_{j_M} purported to be $v_{j_v}(\boldsymbol{\beta})$, and $\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M}\boldsymbol{\lambda}(\boldsymbol{\beta})$, respectively, and $\mathcal{V}$ is left with checking their correctness.

Remark 1. In Π_{GBF2} the polynomials $u_i(\mathbf{X})$ are evaluated at $\boldsymbol{\alpha} \in \mathbb{F}^\nu$ while the $v_i(\mathbf{X})$ ones are evaluated at $\boldsymbol{\beta} \in \mathbb{F}^\nu$. Observe that $u(\boldsymbol{\alpha}) = \mathbf{u}^\top \boldsymbol{\lambda}(\boldsymbol{\alpha}) = u \Leftrightarrow \sum_{\mathbf{h}\in\mathbb{H}} \mathbf{u}^\top \boldsymbol{\lambda}(\mathbf{h}) \cdot \Lambda(\boldsymbol{\alpha},\mathbf{h}) = u \Leftrightarrow \sum_{\mathbf{h}\in\mathbb{H}} p(\mathbf{h}) = u$, where $p(\mathbf{X}) \coloneqq u(\mathbf{X})\Lambda(\boldsymbol{\alpha},\mathbf{X})$. Running the sumcheck argument on this claim reduces to the correctness of evaluation $u(\boldsymbol{\alpha}')$ for a fresh $\boldsymbol{\alpha}' \in \mathbb{F}^\nu$. Hence, if during the second sumcheck (proving equation 16) we include in the batch the claims about correctness of $u_i(\boldsymbol{\alpha})$ we are left with validating the correctness of $u_i(\mathbf{X})$ and $v_i(\mathbf{X})$ at the same point $\boldsymbol{\beta} \in \mathbb{F}^\nu$.

We note that Π_{GBF} can be extended to an interactive reduction from the multi-instance relation $\mathcal{R}^*_{\mathsf{GBF}}$ to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$. The only difference is that the first sumchecks across all the $\mathcal{R}_{\mathsf{GBF}}$ statements need to be batched.

Now, for $\mathcal{V}$ to verify a statement in $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ it needs to verify the correctness of evaluations of multiple committed polynomials. Given a PC scheme equipped with an algorithm PC.Open for proving evaluations of multiple committed polynomials we can reduce the statement to the validity of the evaluation proof. That is, PC.Open can be seen as a reduction from $\mathcal{R}^*_{\mathsf{PCE}}$ to the relation of *polynomial commitment evaluation proof* $\mathcal{R}_{\mathsf{PCEP}}$ which has an empty index and witness, and the instance includes the commitments, the evaluation point, claimed evaluations, and the evaluation proof. We call such an interactive reduction of knowledge Π_{provePCE}. Hence, after $\mathcal{P}$ has reduced $\mathcal{R}_{\mathsf{GBF}}$ to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$, it can interact with $\mathcal{V}$ according to Π_{provePCE} to reduce the latter to $\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$, i.e. the correctness of a batched evaluation proof and batched matrix evaluations, which are both relations with empty witnesses. This leads to a *stateless statement* that can be verified directly.

Finally, we define the special cases of Π_{GBF}, with initial relations $\mathcal{R}_{\mathsf{GBF},\alpha}$ and $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ as $\Pi_{\mathsf{GBF},\alpha}$ and $\Pi_{\mathsf{GBF},\alpha,\beta}$, respectively. Observe that if we run $\Pi_{\mathsf{GBF},\alpha,\beta}$ for the multi-instance relation $\mathcal{R}^*_{\mathsf{GBF},\alpha,\beta}$ then we get a reduction from multiple evaluations of the matrix polynomials to a single, fresh one, as explained at the end of the previous section. This will be important for our PCD construction that we outline next.

2.3 Proof-Carrying Data Overview

In this section we describe how we construct proof-carrying data schemes for bounded-depth computations. In contrast to recursion through folding techniques where the prover needs the previous witness to continue the recursion, our prover requires only the previous *public accumulator*, consisting of (1) a polynomial evaluation proof and (2) claimed holographic checks (i.e. matrix polynomial evaluation claims). Compared to full recursion and atomic accumulation schemes for polynomial evaluations, our prover does not need to compute a full SNARK proof at each step, but merely one that satisfies *proof succinctness*. This is because it does not need to prove correctness of the holographic (public) checks but rather simply state them.

PCD Construction. We design our scheme modularly, using our constructions (i) $\Pi_{\mathsf{GBF}} : \mathcal{R}_{\mathsf{GBF}} \to \mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$, (ii) $\Pi_{\mathsf{batchM}} : \mathcal{R}_{\mathsf{hbPCE}} \to \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$, and assuming (iii) a 2ν-variate polynomial commitment scheme PC_2 and a straightline extractable ν-variate one PC, and generic interactive reductions of knowledge (iv) $\Pi_{\mathsf{provePCE}} : \mathcal{R}^*_{\mathsf{PCE}} \to \mathcal{R}_{\mathsf{PCEP}}$, (v) $\Pi_{\mathsf{batchPCEP}} : \mathcal{R}^*_{\mathsf{PCEP}} \to \mathcal{R}_{\mathsf{PCEP}}$.
The overview of the construction follows. Given ingredients (i) - (iv) we construct an interactive reduction of knowledge Barebones from $\mathcal{R}_{\mathsf{CCS}}$ to $\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$. Barebones captures the main workflow of virtually all argument of knowledge constructions for $\mathcal{R}_{\mathsf{CCS}}$. In particular, when Π_{GBF} is instantiated with Π_{GBF1} and $\nu = 1$ we recover SuperMarlin, while when instantiated with Π_{GBF2} and $\nu = \log n$ we recover SuperSpartan. We define $\mathcal{R}_{\mathsf{Acc}} := \mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ and prove the following theorem.

Theorem 1 (informal). *Given a* PC *and* PC_2 *that satisfy item (iii), there exists a family of (public-coin, indexed) interactive reductions of knowledge* Barebones *from* $\mathcal{R}_{\mathsf{CCS}}$ *to* $\mathcal{R}_{\mathsf{Acc}}$.

Proof (sketch). We start by constructing an interactive reduction of knowledge from $\mathcal{R}_{\mathsf{CCS}}$ to $\mathcal{R}_{\mathsf{GBF},\alpha}$. Leaving aside the public inputs to the circuit for ease of exposition (see remark 4), the witness of $\mathcal{R}_{\mathsf{CCS}}$ is a vector $\mathbf{z} \in \mathbb{F}^n$, such that

$$\sum_{i=1}^{q} c_i \bigcirc_{j \in S_i} \mathbf{M}_j \mathbf{z} = \mathbf{0}$$

where $q, c_i, S_i, \mathbf{M}_j$ specify the CCS structure (i.e. the circuit). Let $\mathcal{P}$ commit to the polynomial $z(\mathbf{X}) = \mathbf{z}^\top \boldsymbol{\lambda}(\mathbf{X})$ and send it to $\mathcal{V}$. Then, given verifier randomness $\boldsymbol{\alpha} \in \mathbb{F}^\nu$, the CCS equation satisfiability is implied (with overwhelming probability) as long as $\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \left(\sum_{i=1}^{q} c_i \bigcirc_{j \in S_i} \mathbf{M}_j \mathbf{z}\right) = 0$. That is, the statement of membership in $\mathcal{R}_{\mathsf{CCS}}$ is reduced to that of $\mathcal{R}_{\mathsf{GBF},\alpha}$. We call this interactive reduction of knowledge Π_{Collapse}.

Then, $\mathcal{P}$ and $\mathcal{V}$ run $\Pi_{\mathsf{GBF},\alpha}$ to reduce the statement to one in $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$. Afterwards, they run the Π_{batchM} protocol to batch the matrix evaluations reducing the statement to one in $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$. They conclude the interaction by batching the polynomial commitment evaluation claims into a single evaluation proof claim by running Π_{provePCE} and reducing to a statement in $\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$.

Then, using the composition properties of interactive reductions of knowledge we define the reduction Barebones which satisfies the properties of Theorem 1 as

$$(\Pi_{\mathsf{provePCE}} \times \mathsf{ID}_{\mathcal{R}_{\mathsf{GBF},\alpha,\beta}}) \circ (\mathsf{ID}_{\mathcal{R}_{\mathsf{PCE}}} \times \Pi_{\mathsf{batchM}}) \circ \Pi_{\mathsf{GBF},\alpha} \circ \Pi_{\mathsf{Collapse}} : \mathcal{R}_{\mathsf{CCS}} \to \mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}.$$

□

Next, using ingredients (i)-(iii), (v) we construct a family of many-to-one interactive reductions of knowledge Π_{Fold} from $\mathcal{R}^*_{\mathsf{Acc}}$ to $\mathcal{R}_{\mathsf{Acc}}$. Intuitively, Π_{Fold} can be seen as a reduction from K claims of the form $\sum_{i \in [t_M]} \eta_i^{(j)} M_i(\boldsymbol{\beta}^{(j)}, \boldsymbol{\alpha}^{(j)}) = \gamma^{(j)}$ where $j \in [K]$ to a single claim $\sum_{i \in [t_M]} \eta_i M_i(\boldsymbol{\beta}, \boldsymbol{\alpha}) = \gamma$ for fresh $\boldsymbol{\eta}, \boldsymbol{\alpha}, \boldsymbol{\beta}$, and from K' polynomial commitment evaluation proof claims to a single one. We prove the following theorem.

Theorem 2 (informal). *Given a* PC *and* PC_2 *that satisfy item (iii), there exists a family of (public-coin, indexed) interactive reductions of knowledge* Π_{Fold} *from* $\mathcal{R}^*_{\mathsf{Acc}}$ *to* $\mathcal{R}_{\mathsf{Acc}}$.

Proof (sketch). On input multiple statements in $\mathcal{R}_{\mathsf{Acc}} = \mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$, $\mathcal{P}$ and $\mathcal{V}$ start by running the $\Pi_{\mathsf{GBF},\alpha,\beta}$ protocol, reducing the many $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ statements to ones in $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$. Then, they invoke Π_{provePCE} and Π_{batchM} in parallel in order to reduce the polynomial commitment evaluation claims to correctness of a single evaluation proof and to batch the claimed matrix evaluations. Finally, they run the $\Pi_{\mathsf{batchPCEP}}$ in order to batch the many evaluation proof statements to a single one, i.e. to reduce $\mathcal{R}^*_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ to $\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$.

Using the composition properties of interactive reductions of knowledge, we define the reduction Π_{Fold} which satisfies the properties of Theorem 2 as

$$(\Pi_{\mathsf{batchPCEP}} \times \mathsf{ID}_{\mathcal{R}_{\mathsf{GBF},\alpha,\beta}}) \circ (\mathsf{ID}_{\mathcal{R}_{\mathsf{PCEP}}} \times \Pi_{\mathsf{provePCE}} \times \Pi_{\mathsf{batchM}}) \circ (\mathsf{ID}_{\mathcal{R}_{\mathsf{PCEP}}} \times \Pi_{\mathsf{GBF},\alpha,\beta}) : \mathcal{R}^*_{\mathsf{Acc}} \to \mathcal{R}_{\mathsf{Acc}}.$$

□

Compiling Barebones *and* Π_{Fold} *into PCD.* Prior works [BCMS20, Thm. 5.2], [BCL+21, Thm. 5.3] have constructed polynomial-time transformations that given a non-interactive argument of knowledge ARG and a non-interactive accumulation scheme ACC for ARG with sublinear verification, both in the standard model, give a standard model proof-carrying data scheme for bounded-depth computations. Intuitively, an accumulation scheme for an argument of knowledge is a many-to-one reduction for the relation capturing the satisfiability of the argument's verification predicate. We describe how to get the necessary components to apply the transformations of the theorems.

Since the constructions of Theorem 1 and Theorem 2 are public-coin, we can apply the Fiat-Shamir transform [FS87] and obtain non-interactive versions NI-Barebones and NI-Π_{Fold} in the ROM. Then, we use the polynomial-time transformation of [BMNW25, Thm. 4.3] to obtain a pair of protocols in the random oracle model, ARG and ACC, such that ARG is a non-interactive argument for $\mathcal{R}_{\mathsf{CCS}}$ and ACC is an accumulation scheme for ARG. Finally, by heuristically instantiating the oracle with an appropriate hash function

we obtain a non-interactive argument and accumulation scheme in the standard model with heuristic security. Since the verifiers of both Barebones and Π_{Fold} are succinct, our accumulation verifier is as well.

In Fig. 1 we give a pictorial representation of the augmented circuit that realizes recursion. With dashed red boxes we indicate the extra ingredients required for accumulating the public structure statements. The rest of the circuit realizes the accumulation of the polynomial commitment evaluation proofs. In the case of full recursion, $\mathsf{acc}^{\mathsf{PCEP}}$ is empty and $\mathcal{V}_{\mathsf{batchPCEP}}$ is the evaluation proof verifier of PC.

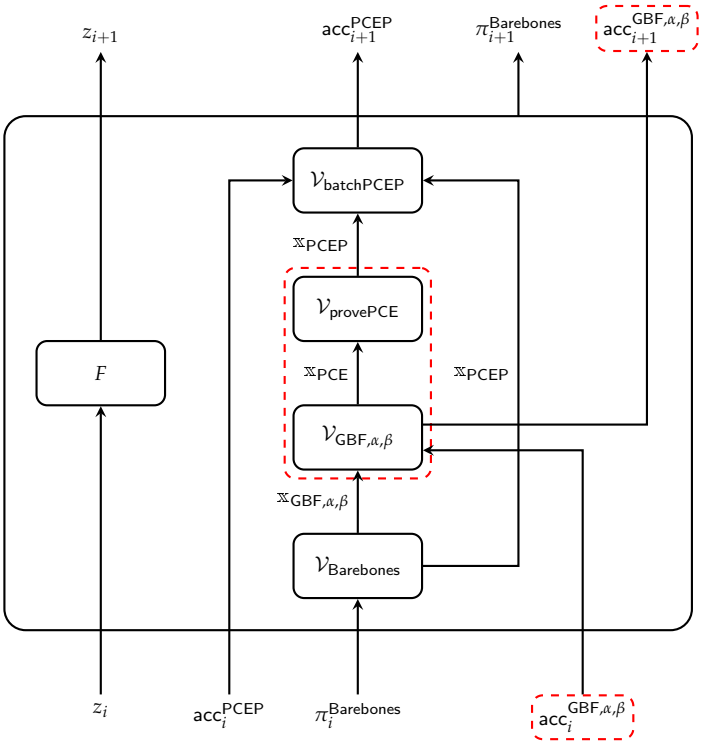

Fig. 1. Augmented recursive circuit combining PC and holography accumulation.

Decider for Non-Uniform PCD. We describe now how to efficiently decide the final output of a non-uniform PCD scheme by assuming a homomorphic polynomial commitment scheme PC_2 such as the ones in [BMM+21, GPPS24, NPR24]. To this end, let $\mathcal{F} = \{f_1, \ldots, f_\ell\}$ be a set of ℓ functions used in the distributed computation and assume w.l.o.g. that each function is encoded in CCS using t_M matrices (not necessarily the same ones for all of the functions). Following the SuperNova [KS23] blueprint we assume that an accumulator acc at every point of the recursion is a vector of size ℓ of accumulators $\mathsf{acc}^{(i)}$ corresponding to function f_i.

In order to check the validity of the non-uniform PCD, one needs to check the final output which consists of (1) an argument proof, and (2) an accumulator. The former consists of (1a) a statement in $\mathcal{R}_{\mathsf{PCEP}}$, and (1b) a statement in $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$, while the latter consists of (2a) ℓ statements in $\mathcal{R}_{\mathsf{PCEP}}$, and (2b) ℓ statements in $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$. Instead of having the decider check each statement individually, we allow it to interact with the prover as follows.

First, the prover and decider run $\Pi_{\mathsf{GBF},\alpha,\beta}$ to reduce the $\ell + 1$ statements of $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ to $\mathcal{R}_{\mathsf{hbPCE}}$, followed by Π_{batchM} to reduce the latter to a statement in $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$. Let $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{F}^\nu$ be the evaluation

points defined in the invocation of $\Pi_{\mathsf{GBF},\alpha,\beta}$, $\boldsymbol{\eta} \in \mathbb{F}^{\ell \cdot t_M}$ be the coefficients in the final $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ instance defined during Π_{batchM}, and $\gamma \in \mathbb{F}$ the claimed value. Then, the prover computes the polynomial $M(\mathbf{X}, \mathbf{Y}) = \sum_{i \in [\ell]} \sum_{j \in [t_M]} \eta_j^{(i)} M_j^{(i)}(\mathbf{X}, \mathbf{Y})$ and sends an evaluation proof for the claim $M(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \gamma$. The decider derives the commitment $[M(\mathbf{X}, \mathbf{Y})]$ by linearly combining the matrix commitments. Moreover, they run Π_{provePCE} to reduce the $\mathcal{R}_{\mathsf{PCE}}$ statements produced by $\Pi_{\mathsf{GBF},\alpha,\beta}$ to a statement in $\mathcal{R}_{\mathsf{PCEP}}$, followed by $\Pi_{\mathsf{batchPCEP}}$ to reduce the $\ell + 2$ statements in $\mathcal{R}_{\mathsf{PCEP}}$ to a single one. In the end, the decider checks the final matrix evaluation proof and final polynomial commitment evaluation proof.

In total, the decider cost is dominated by a single linear combination of size equal to the total number of matrices and verifying a single evaluation proof. This is in contrast to state-of-the-art deciders such as the one in [BHKZ25] that require checking $\ell \cdot t_M$ matrix polynomial evaluations.

3 Definitions and Building Blocks

Definition 1. *An **indexed relation** $\mathcal{R}$ is a set of triples $\{(\mathbb{i}, \mathbb{x}, \mathbb{w})\}$, where $\mathbb{i}$ is the index, $\mathbb{x}$ is the instance, and $\mathbb{w}$ is the witness. The corresponding **indexed language** $\mathcal{L}(\mathcal{R})$ is the set of pairs $\{(\mathbb{i}, \mathbb{x})\}$ for which there exists $\mathbb{w}$ such that $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}$. A **relation relative to a random oracle**, denoted $\mathcal{R}^{\mathcal{U}}$, is a set of relations $\{\mathcal{R}^{\rho} : \rho \in \mathrm{supp}(\mathcal{U})\}$, where $\mathrm{supp}(\mathcal{U})$ denotes $\bigcup_{\lambda \in \mathbb{N}} \mathrm{supp}(\mathcal{U}(1^{\lambda}))$. A **parametrized indexed relation** over a parameter space $\mathbb{P}$ is a set of relations $\{\mathcal{R}(\mathsf{pp}) : \mathsf{pp} \in \mathbb{P}\}$. The **truth relation** is defined to be $\mathcal{R}_{\top} := (\bot, \mathsf{true}, \bot)$. The **multi-instance relation** is defined to be*

$$\mathcal{R}^* := \left\{ \left(\mathbb{i}, \left(\mathbb{x}^{(1)}, \ldots, \mathbb{x}^{(m)} \right), \left(\mathbb{w}^{(1)}, \ldots, \mathbb{w}^{(m)} \right) \right) \middle| \forall i \in [m], \left(\mathbb{i}, \mathbb{x}^{(i)}, \mathbb{w}^{(i)} \right) \in \mathcal{R} \right\}.$$

We use the definitions of interactive reductions of knowledge from [KP23] adapted to work for indexed relations and replacing knowledge soundness with round-by-round knowledge soundness from [ACFY24].

Definition 2. *Let $\mathcal{R}$ and $\mathcal{R}'$ be parametrized indexed relations (over the same parameter set). A **(public-coin, indexed) interactive reduction of knowledge** from $\mathcal{R}$ to $\mathcal{R}'$ is a tuple of polynomial-time algorithms $\Pi = (\mathcal{G}, \mathcal{I}, \mathcal{P}, \mathcal{V})$ with the following syntax.*

- *The generator $\mathcal{G}$ receives as input a security parameter λ (in unary) and outputs public parameters pp.*
- *The indexer $\mathcal{I}$ is a deterministic algorithm which receives as input public parameters pp, and an index $\mathbb{i}$ (of $\mathcal{R}$). It outputs short index ι, and a new index $\mathbb{i}'$ (of $\mathcal{R}'$).*
- *The prover $\mathcal{P}$ is an interactive algorithm that takes as input public parameters pp, the index $\mathbb{i}$, an instance $\mathbb{x}$, and a witness $\mathbb{w}$. It interacts with $\mathcal{V}$ over k rounds, where in each round i it sends message π_i and reduces the statement $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}$ to a new statement $(\mathbb{i}', \mathbb{x}', \mathbb{w}') \in \mathcal{R}'$.*
- *The verifier $\mathcal{V}$ is an interactive algorithm that takes as input the short index ι, and instance $\mathbb{x}$. It interacts with $\mathcal{P}$ over k rounds, where in each round i it sends random element $r_i \in \mathbb{F}$ and reduces the task of checking $(\mathbb{i}, \mathbb{x}) \in \mathcal{L}(\mathcal{R})$ to the task of checking a new statement $(\mathbb{i}', \mathbb{x}') \in \mathcal{L}(\mathcal{R}')$.*

Without loss of generality, $\mathcal{V}$ is split into three phases. In the interaction phase it samples random elements and sends them to $\mathcal{P}$. In the decision phase it performs deterministic checks on $\mathcal{P}$'s messages, accepting or rejecting the interaction. If $\mathcal{V}$ accepts the interaction, in the output phase it outputs $\mathbb{x}'$.

Completeness. *Π is perfectly complete if the following holds. For every adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{c|l} (\mathbb{i}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}(\mathsf{pp}) & \mathsf{pp} \leftarrow \mathcal{G}(1^{\lambda}) \\ \Downarrow & (\mathbb{i}, \mathbb{x}, \mathbb{w}) \leftarrow \mathcal{A} \\ (\mathbb{i}', \mathbb{x}', \mathbb{w}') \in \mathcal{R}'(\mathsf{pp}) & (\iota, \mathbb{i}') \leftarrow \mathcal{I}(\mathsf{pp}, \mathbb{i}) \\ & (\mathbb{x}', \mathbb{w}') \leftarrow \langle \mathcal{P}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathbb{w}), \mathcal{V}(\iota, \mathbb{x}) \rangle \end{array}\right] = 1.$$

State function. *A state function* State *for* Π *is a (possibly inefficient) function that takes as inputs* $\mathsf{pp}, \mathbb{i}, \mathbb{x}$*, and a transcript* tr *and outputs a bit, for which the following hold.*

- ***Empty transcript:*** *If* $\mathsf{tr} = \emptyset$ *is the empty set, then* $\mathsf{State}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathsf{tr}) = 1$ *iff* $(\mathbb{i}, \mathbb{x}) \in \mathcal{L}(\mathcal{R})$.
- ***Prover moves:*** *If* tr *is a transcript where the prover is about to move, and* $\mathsf{State}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathsf{tr}) = 0$ *then, for any prover message* π, $\mathsf{State}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathsf{tr}||\pi) = 0$.
- ***Full transcript:*** *If* tr *is a full transcript and* $\mathsf{State}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathsf{tr}) = 0$*, then* $\mathcal{V}$ *either rejects the interaction in the decision phase, or outputs* $\mathbb{x}'$ *such that* $(\mathbb{i}', \mathbb{x}') \notin \mathcal{L}(\mathcal{R}')$.

Round-by-round knowledge soundness. *A k-round* Π *has* ***round-by-round knowledge soundness*** *with errors* $(\epsilon_1, \ldots, \epsilon_k)$ *and extraction time* et *if* Π *has a state function* State *and there exists a deterministic extractor* $\mathcal{E}$ *that runs in time at most* et *such that the following holds: for every instance* $\mathbb{x}$ *and transcript* $\mathsf{tr} = (\pi_1, r_1, \ldots, \pi_{i-1}, r_{i-1}, \pi_i)$ *if*

$$(i)\ \mathsf{State}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathsf{tr}) = 0, \text{ and} \qquad (ii)\ \Pr_{r_i}[\mathsf{State}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathsf{tr}||r_i) = 1] > \epsilon_i,$$

then for any transcript continuation $\mathsf{tr}' = (\pi_{i+1}, r_{i+1}, \ldots, \pi_k, r_k)$ *the extractor outputs* $\mathbb{w} \leftarrow \mathcal{E}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathsf{tr}||r_i||\mathsf{tr}', \mathbb{w}')$ *such that* $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}$*. If* et *is unbounded then we omit the word "knowledge".*

Public reducibility. *There exists a polynomial-time function* ϕ *such that for any* PPT *adversaries* $\mathcal{A}, \tilde{\mathcal{P}}$*, given* $\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda)$, $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \leftarrow \mathcal{A}(\mathsf{pp})$, $(\iota, \mathbb{i}') \leftarrow \mathcal{I}(\mathsf{pp}, \mathbb{i})$ *and* $(\mathbb{x}', \mathbb{w}') \leftarrow \langle \tilde{\mathcal{P}}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathbb{w}), \mathcal{V}(\iota, \mathbb{x}) \rangle$ *with interaction transcript* tr*, we have that* $\phi(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \mathsf{tr}) = \mathbb{x}'$.

We study several efficiency characteristics of an interactive reduction of knowledge Π*, namely: (i) round complexity,* rc_Π*, (ii) messages sent by the prover,* msp_Π*, (iii) verifier random elements,* vre_Π*, and (iv) verifier complexity,* vc_Π.

We state the following theorem from [KS24] proving that the class of interactive reductions of knowledge is closed under sequential and parallel composition.

Theorem 3 ([KS24]). *Let* $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$ *be indexed relations. Let* $\Pi_1 : \mathcal{R}_1 \to \mathcal{R}_2$, $\Pi_2 : \mathcal{R}_2 \to \mathcal{R}_3$*, and* $\Pi_3 : \mathcal{R}_3 \to \mathcal{R}_4$ *be interactive reductions of knowledge. Then,* $\Pi_2 \circ \Pi_1$ *is an interactive reduction of knowledge from* $\mathcal{R}_1$ *to* $\mathcal{R}_3$ *and* $\Pi_1 \times \Pi_3$ *is an interactive reduction of knowledge from* $\mathcal{R}_1 \times \mathcal{R}_3$ *to* $\mathcal{R}_2 \times \mathcal{R}_4$.

We will use the definition of straightline knowledge sound non-interactive reductions of knowledge from [BMNW25] (for non-promise relations).

Definition 3. *Let* $\mathcal{R}_1^{\mathcal{U}}$ *and* $\mathcal{R}_2^{\mathcal{U}}$ *be parameterized indexed relations (relative to a random oracle). A (preprocessing) non-interactive reduction from* $\mathcal{R}_1$ *to* $\mathcal{R}_2$ *in the random oracle model is a tuple of polynomial-time algorithms* $\mathsf{NI}\text{-}\Pi = (\mathcal{G}, \mathcal{I}, \mathcal{P}, \mathcal{V})$*, of which* $\mathcal{I}, \mathcal{P}, \mathcal{V}$ *have access to the same random oracle, with the following syntax.*

- *The generator* $\mathcal{G}$ *receives as input a security parameter* λ *(in unary) and outputs public parameters* pp.
- *The indexer* $\mathcal{I}$ *is a deterministic algorithm which receives as input public parameters* pp *and index* $\mathbb{i}$*, and outputs a proving key* pk*, verification key* vk*, and new index* $\mathbb{i}'$.
- *The prover* $\mathcal{P}$ *receives as input proving key* pk*, an instance* $\mathbb{x}$*, and witness* $\mathbb{w}$*, and outputs a proof* π *and new witness* $\mathbb{w}'$.
- *The verifier* $\mathcal{V}$ *is a deterministic algorithm which receives as input verification key* vk*, instance* $\mathbb{x}$*, and proof* π*, and outputs a new instance* $\mathbb{x}'$.

Completeness. $\mathsf{NI}\text{-}\Pi$ *is complete if the following holds. For every adversary* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c|l} (\mathbb{i},\mathbb{x},\mathbb{w}) \in \mathcal{R}_1^\rho(\mathrm{pp}) & \rho \leftarrow \mathcal{U}(\lambda) \\ \Downarrow & \mathrm{pp} \leftarrow \mathcal{G}(1^\lambda) \\ (\mathbb{i}',\mathbb{x}',\mathbb{w}') \in \mathcal{R}_2^\rho(\mathrm{pp}) & (\mathbb{i},\mathbb{x},\mathbb{w}) \leftarrow \mathcal{A}^\rho(\mathrm{pp}) \\ & (\mathsf{pk},\mathsf{vk},\mathbb{i}') \leftarrow \mathcal{I}^\rho(\mathrm{pp},\mathbb{i}) \\ & (\pi,\mathbb{w}') \leftarrow \mathcal{P}^\rho(\mathsf{pk},\mathbb{x},\mathbb{w}) \\ & \mathbb{x}' \leftarrow \mathcal{V}^\rho(\mathsf{vk},\mathbb{x},\pi) \end{array}\right] = 1.$$

Straightline knowledge soundness. $\mathsf{NI}\text{-}\Pi$ *is straightline knowledge sound (with respect to auxiliary input distribution $\mathcal{D}$) if the following holds. There exists a deterministic polynomial-time extractor $\mathcal{E}$ such that for every (non-uniform) polynomial-time adversary $\tilde{\mathcal{P}}$,*

$$\Pr\left[\begin{array}{c|l} (\mathbb{i}',\mathbb{x}',\mathbb{w}') \in \mathcal{R}_2^\rho(\mathrm{pp}) & \rho \leftarrow \mathcal{U}(\lambda) \\ \wedge & \mathrm{pp} \leftarrow \mathcal{G}(1^\lambda) \\ (\mathbb{i},\mathbb{x},\mathbb{w}) \notin \mathcal{R}_1^\rho(\mathrm{pp}) & \mathrm{ai} \leftarrow \mathcal{D}(1^\lambda) \\ & (\mathbb{i},\mathbb{x},\pi,\mathbb{w}';\mathrm{tr}) \leftarrow \tilde{\mathcal{P}}^\rho(\mathrm{pp},\mathrm{ai}) \\ & (\mathsf{pk},\mathsf{vk},\mathbb{i}') \leftarrow \mathcal{I}^\rho(\mathrm{pp},\mathbb{i}) \\ & \mathbb{x}' \leftarrow \mathcal{V}^\rho(\mathsf{vk},\mathbb{x},\pi) \\ & \mathbb{w} \leftarrow \mathcal{E}(\mathrm{pp},\mathbb{i},\mathbb{x},\pi,\mathbb{w}',\mathrm{ai},\mathrm{tr}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

As part of our PCD construction we will use the following compiler from [BMNW25] which, given specific non-interactive reductions of knowledge as input, outputs an argument of knowledge and an accumulation scheme for it (see Definitions 10 and 11).

Theorem 4 ([BMNW25]). *There exists a polynomial-time transformation* T *such that the following holds. Let $\mathcal{R}$ be a parameterized indexed relation. Let $\mathcal{R}_{\mathsf{ACC}}^{\mathcal{U}}$ be a parameterized indexed promise relation in $\mathrm{NP}^{\mathcal{U}}$ with the same parameter space as $\mathcal{R}$. Suppose we are given the following non-interactive reductions in the random oracle model:*

- $\mathsf{NI}\text{-}\Pi_{\mathsf{Cast}} = (\mathcal{G}_{\mathsf{Cast}}, \mathcal{I}_{\mathsf{Cast}}, \mathcal{P}_{\mathsf{Cast}}, \mathcal{V}_{\mathsf{Cast}})$*, a reduction from $\mathcal{R}$ to $\mathcal{R}_{\mathsf{ACC}}$.*
- $\mathsf{NI}\text{-}\Pi_{\mathsf{Fold}} = (\mathcal{G}_{\mathsf{Fold}}, \mathcal{I}_{\mathsf{Fold}}, \mathcal{P}_{\mathsf{Fold}}, \mathcal{V}_{\mathsf{Fold}})$*, a many-to-one reduction from $\mathcal{R}_{\mathsf{ACC}}^*$ to $\mathcal{R}_{\mathsf{ACC}}$ with the same generator algorithm as $\mathsf{NI}\text{-}\Pi_{\mathsf{Cast}}$ (i.e., $\mathcal{G}_{\mathsf{Fold}} \equiv \mathcal{G}_{\mathsf{Cast}}$).*

Then $\mathsf{T}[\mathsf{NI}\text{-}\Pi_{\mathsf{Cast}}, \mathsf{NI}\text{-}\Pi_{\mathsf{Fold}}, \mathcal{R}_{\mathsf{ACC}}] = (\mathsf{ARG}, \mathsf{ACC})$, where ARG is a non-interactive argument for $\mathcal{R}$ and ACC is an accumulation scheme for ARG, both in the random oracle model.

We define auxiliary relations whose parameter space is the commitment key space of a polynomial commitment scheme (see Definition 9). We assume a commitment key ck includes the corresponding verification key vk.

Definition 4. *Let PC be a ν-variate polynomial commitment scheme with commitment key ck. The **polynomial commitment evaluation relation** parametrized by ck, denoted $\mathcal{R}_{\mathsf{PCE}}(\mathsf{ck})$, is the set of triples $(\mathbb{i},\mathbb{x},\mathbb{w})$, where*

$$\mathbb{i} = \bot, \quad \mathbb{x} = (\mathbf{d}, [c], \boldsymbol{\alpha}, \beta) \in (\mathbb{N}^\nu \times \mathbb{C} \times \mathbb{F}^\nu \times \mathbb{F}), \quad \mathbb{w} = p(\mathbf{X}) \in \mathbb{F}^{\leq \mathbf{d}}[\mathbf{X}]$$

such that $p(\boldsymbol{\alpha}) = \beta$, and $[c] = \mathsf{PC.Commit}(\mathsf{ck}, p(\mathbf{X}), \mathbf{d})$.

*The **polynomial commitment evaluation proof relation** parametrized by ck, denoted $\mathcal{R}_{\mathsf{PCEP}}(\mathsf{ck})$, is the set of triples $(\mathbb{i},\mathbb{x},\mathbb{w})$, where*

$$\mathbb{i} = \bot, \quad \mathbb{x} = (\mathbf{d}, [c], \boldsymbol{\alpha}, \beta, \pi) \in (\mathbb{N}^\nu \times \mathbb{C} \times \mathbb{F}^\nu \times \mathbb{F} \times \mathbb{P}), \quad \mathbb{w} = \bot$$

such that $\mathsf{PC.Verify}(\mathsf{vk}, \mathbf{d}, [c], \boldsymbol{\alpha}, \beta, \pi) = 1$.

*We say that a PC has a **batch evaluation proof scheme** if there exists an interactive reduction of knowledge $\Pi_{\mathsf{provePCE}} : \mathcal{R}_{\mathsf{PCE}}^* \to \mathcal{R}_{\mathsf{PCEP}}$. Furthermore, we say that PC has **accumulatable evaluation proofs** if there exists an interactive reduction of knowledge $\Pi_{\mathsf{batchPCEP}} : \mathcal{R}_{\mathsf{PCEP}}^* \to \mathcal{R}_{\mathsf{PCEP}}$.*

Definition 5. *Let* PC_2 *be a* 2ν*-variate polynomial commitment scheme with commitment key* ck_2*. The* ***holographic bivariate polynomial commitment evaluation relation*** *parametrized by* ck_2*, denoted* $\mathcal{R}_{\mathsf{hbPCE}}(\mathsf{ck}_2)$*, is the set of triples* $(\mathbb{i}, \mathbb{x}, \mathbb{w})$*, where*

$$\mathbb{i} = (n, d_v, k, \{P_i(\mathbf{X}, \mathbf{Y})\}_{i=1}^{k}) \in (\mathbb{N}^3 \times (\mathbb{F}^{\leq d_v}[\mathbf{X}, \mathbf{Y}])^k),$$
$$\mathbb{x} = (\{[P_i(\mathbf{X}, \mathbf{Y})], \gamma_i\}_{i=1}^{k}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \in ((\mathbb{C} \times \mathbb{F})^k \times \mathbb{F}^\nu \times \mathbb{F}^\nu),\ \mathbb{w} = \bot$$

such that $[P_i(\mathbf{X}, \mathbf{Y})] = \mathsf{PC}_2.\mathsf{Commit}(\mathsf{ck}_2, P_i(\mathbf{X}, \mathbf{Y}), d_v)$ *and* $P_i(\boldsymbol{\beta}, \boldsymbol{\alpha}) = \gamma_i$*,* $\forall i \in [k]$*.*

We recall the customizable constraint system relation [STW23], an algebraic language for the satisfiability of circuits with high-degree gates.

Definition 6 ([STW23]). *The* ***customizable constraint system relation*** *denoted by* $\mathcal{R}_{\mathsf{CCS}}$*, is the set of triples* $(\mathbb{i}, \mathbb{x}, \mathbb{w})$ *where*

- *the index* $\mathbb{i}$ *consists of size bounds* $n, t_M \in \mathbb{N}$*, and a set* $\mathcal{M} = \{\mathbf{M}_1, \ldots, \mathbf{M}_{t_M}\}$ *of* t_M *matrices in* $\mathbb{F}^{n \times n}$*,*
- *the instance* $\mathbb{x}$ *consists of (i) size bounds* $q, d, \ell \in \mathbb{N}$*, (ii) a set of constants* $\{c_1, \ldots, c_q\} \in \mathbb{F}^q$*, (iii) a multiset* $\{S_1, \ldots, S_q\}$*, where each* $S_i \subseteq \{1, \ldots, t_M\}$*, with* $|S_i| \leq d$*, and (iv) a vector* $\mathbf{x} \in \mathbb{F}^\ell$*,*
- *the witness* $\mathbb{w}$ *consists of a vector* $\mathbf{w} \in \mathbb{F}^{n-\ell}$*,*

such that the following holds for $\mathbf{z} = (\mathbf{x}, \mathbf{w})$*:* $\sum_{i=1}^{q} c_i \bigcirc_{j \in S_i} \mathbf{M}_j \mathbf{z} = \mathbf{0}$*.*

4 Generalized Bilinear Forms

In this section we present the *generalized bilinear forms relation* $\mathcal{R}_{\mathsf{GBF}}$, for which we construct a family of interactive reductions of knowledge Π_{GBF} to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$ with different efficiency profiles. The family is split into two protocols, Π_{GBF1} (Section. 4.1) and Π_{GBF2} (Section. 4.2), where the number indicates how many sumcheck arguments are used. Both constructions can be instantiated with either univariate or multivariate polynomials. Looking ahead, in Section 5 we will show how with the different instantiations of Π_{GBF} we can recover SuperMarlin and SuperSpartan [STW23]. Moreover, we will show how it can be paired with a ν-variate PC scheme that has accumulatable evaluations to design PCD schemes.

Definition 7. *The* ***generalized bilinear forms relation*** *parametrized by* $\mathsf{pp} = (\mathsf{ck}, \mathsf{ck}_2)$*, denoted* $\mathcal{R}_{\mathsf{GBF}}(\mathsf{pp})$*, is the set of triples* $(\mathbb{i}, \mathbb{x}, \mathbb{w})$ *where*

- Index. $\mathbb{i}$ *consists of size bounds* $n, t_M \in \mathbb{N}$ *and a set of* $n \times n$ *matrices* $\mathcal{M} = \{\mathbf{M}_1, \ldots, \mathbf{M}_{t_M}\}$*.*
- Instance. $\mathbb{x}$ *consists of (i) size bounds* $(t_u, t_v, q_l, q_r, d_l, d_r) \in \mathbb{N}^6$*; (ii) constants* $\{c_{l,1}, \ldots, c_{l,q_l}\} \in \mathbb{F}^{q_l}$ *and* $\{c_{r,1}, \ldots, c_{r,q_r}\} \in \mathbb{F}^{q_r}$*; (iii) multisets* $\{S_{l,1}, \ldots, S_{l,q_l}\}$ *with* $S_{l,i} \subseteq \{1, \ldots, t_u\}$ *and* $|S_{l,i}| \leq d_l$*, and* $\{S_{r,1}, \ldots, S_{r,q_r}\}$ *with* $S_{r,i} \subseteq \{1, \ldots, t_M\} \times \{1, \ldots, t_v\}$ *and* $|S_{r,i}| \leq d_r$*; (iv)* PC *commitments to* ν*-variate polynomials* $\{[u_i(\mathbf{X})]_{i=1}^{t_u}, [v_i(\mathbf{X})]_{i=1}^{t_v}\}$*, (v)* PC_2 *commitments to* 2ν*-variate polynomials* $\{[M_i(\mathbf{X}, \mathbf{Y})]\}_{i=1}^{t_M}$*; (vi) a field element* $s \in \mathbb{F}$*.*
- Witness. $\mathbb{w}$ *consists of vectors* $\{\mathbf{u}_1, \ldots, \mathbf{u}_{t_u}\}$ *and* $\{\mathbf{v}_1, \ldots, \mathbf{v}_{t_v}\}$ *in* $\mathbb{F}^n$*.*

These satisfy

$$s = \left(\sum_{i=1}^{q_l} c_{l,i} \bigcirc_{j \in S_{l,i}} \mathbf{u}_j\right)^\top \left(\sum_{i=1}^{q_r} c_{r,i} \bigcirc_{(j_M, j_v) \in S_{r,i}} \mathbf{M}_{j_M} \mathbf{v}_{j_v}\right),$$

and $[u_i(\mathbf{X})] = \mathsf{PC}.\mathsf{Commit}(\mathsf{ck}, u_i(\mathbf{X}), d_v)$ *and* $[v_i(\mathbf{X})] = \mathsf{PC}.\mathsf{Commit}(\mathsf{ck}, v_i(\mathbf{X}), d_v)$*. Moreover, let* $S_R \subseteq [t_M] \times [t_v]$ *be the set of pairs* (j_M, j_v) *such that* $\exists i \in [q_r]$ *with* $(j_M, j_v) \in S_{r,i}$*, and define* $t_{M,v} = |S_R|$*.*

We define two special cases of $\mathcal{R}_{\mathsf{GBF}}$ for when the component vectors are the evaluations of the Lagrange basis.

Definition 8. *Let $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{F}^{\nu}$. We define $\mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha}} \subseteq \mathcal{R}_{\mathsf{GBF}}$ such that $t_u = q_l = d_l = c_{l,1} = 1$, $S_{l,1} = \{1\}$, and $\mathbf{u} = \boldsymbol{\lambda}(\boldsymbol{\alpha})$. Moreover, we define $\mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}} \subseteq \mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha}}$ if in addition $t_v = d_r = 1$, $q_r = t_M$, and $S_{r,i} = \{(i,1)\}$, $\forall i \in [q_r]$.*

Remark 2. In the relations $\mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha}}$ and $\mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}$ we replace the commitments to $u(\mathbf{X}) = \boldsymbol{\lambda}(\boldsymbol{\alpha})^{\top}\boldsymbol{\lambda}(\mathbf{X})$ and $v(\mathbf{X}) = \boldsymbol{\lambda}(\boldsymbol{\beta})^{\top}\boldsymbol{\lambda}(\mathbf{X})$ with the field elements $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, respectively, since these serve as implicit commitments to the vectors. Moreover, the witness of the relation $\mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}$ is empty.

Remark 3. Unless stated otherwise our constructions assume a 2ν-variate polynomial commitment scheme PC_2 and an extractable ν-variate polynomial commitment scheme PC with extractor $\mathcal{E}_{\mathsf{PC}}$, that has accumulatable evaluations and a batch evaluation proof scheme. Moreover, each generator $\mathcal{G}$ on input 1^{λ} runs $\mathsf{PC}.\mathsf{Setup}(1^{\lambda}, \mathbf{D})$ and $\mathsf{PC}_2.\mathsf{Setup}(1^{\lambda}, \mathbf{D}')$ to get $(\mathsf{pp}_{\mathsf{PC}}, \mathsf{pp}_{\mathsf{PC}_2})$ and outputs $\mathsf{pp} := (\mathsf{ck}, \mathsf{ck}_2)$, where $\mathsf{ck} \leftarrow \mathsf{PC}.\mathsf{Trim}^{\mathsf{pp}_{\mathsf{PC}}}(1^{\lambda}, d_{\nu})$, $\mathsf{ck}_2 \leftarrow \mathsf{PC}_2.\mathsf{Trim}^{\mathsf{pp}_{\mathsf{PC}_2}}(1^{\lambda}, d'_{\nu})$, and $d_1 = d'_1 = n-1$, $d_{\log n} = d'_{\log n} = 1$.

We state and prove the following lemma which constructs an auxiliary reduction from $\mathcal{R}_{\mathsf{hbPCE}}$ to $\mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}$. Intuitively, the reduction is simply the linear combination of the claimed evaluations of the matrix polynomials.

Lemma 1. *There exists an interactive reduction of knowledge Π_{batchM} from $\mathcal{R}_{\mathsf{hbPCE}}$ to $\mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}$, with perfect completeness, round-by-round knowledge soundness error $\frac{1}{|\mathbb{F}|}$, and the following efficiency properties: (i) round complexity: 1, (ii) messages sent by $\mathcal{P}$: 0, (iii) verifier random elements: t_M, and (iv) verification complexity: $\mathcal{O}(t_M)$.*

Proof. The indexer $\mathcal{I}$ on input the public parameters pp and index $\mathbb{i}_{\mathsf{hbPCE}}$, computes $\forall k \in [t_M]$, matrix $\mathbf{M}_k$ such that $\mathbf{M}_k^{i,j} = P_k(\mathbf{h}_i, \mathbf{h}_j)$, $\forall i,j \in [n]$. It outputs short index $\iota = (n, t_M)$ and index $\mathbb{i}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}} = (n, t_M, \mathcal{M})$. The interaction is as follows. $\mathcal{V}$ picks $\boldsymbol{c} \in \mathbb{F}^{t_M}$ and sends it to $\mathcal{P}$. Then, $\mathcal{P}$ and $\mathcal{V}$ compute and output the $\mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}$ instance $\mathbb{x}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}$, such that $\mathbb{x}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}.s = \sum_{i\in[t_M]} c_i \cdot \mathbb{x}_{\mathsf{hbPCE}}.\gamma_i$ and $\mathbb{x}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}.[M_i(\mathbf{X},\mathbf{Y})] = \mathbb{x}_{\mathsf{hbPCE}}.[P_i(\mathbf{X},\mathbf{Y})]$. $\mathcal{P}$ outputs $\mathbb{w}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}} = \perp$.

Completeness of the reduction is immediate. For knowledge soundness, since the witness is empty, it suffices to prove soundness. To do this, observe that if $(\mathbb{i}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}, \mathbb{x}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}, \perp) \in \mathcal{R}_{\mathsf{GBF},\boldsymbol{\alpha},\boldsymbol{\beta}}$ then

$$\sum_{i\in[t_M]} c_i \cdot \mathbb{x}_{\mathsf{hbPCE}}.\gamma_i = s = \sum_{i\in[t_M]} c_i \boldsymbol{\lambda}(\boldsymbol{\alpha})^{\top}\mathbf{M}_i\boldsymbol{\lambda}(\boldsymbol{\beta})$$

Hence, except with probability $\frac{1}{|\mathbb{F}|}$ we have that $\forall i \in [t_M]$, $\gamma_i = \boldsymbol{\lambda}(\boldsymbol{\alpha})^{\top}\mathbf{M}_i\boldsymbol{\lambda}(\boldsymbol{\beta}) = M_i(\boldsymbol{\beta},\boldsymbol{\alpha})$, concluding the proof. □

□

4.1 Π_{GBF1}

In this section we present the interactive reduction of knowledge Π_{GBF1}. The protocol uses a single invocation of the sumcheck argument, similar to the Marlin [CHM+20] approach for proving R1CS satisfiability.

Theorem 5. *The protocol Π_{GBF1} of Fig. 2, 3 is an interactive reduction of knowledge from $\mathcal{R}^*_{\mathsf{GBF}}$ to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$ with perfect completeness. with perfect completeness. Its efficiency and round-by-round knowledge-soundness properties are summarized in Table 1.*

Proof. The indexer $\mathcal{I}$ on input public parameters pp and index $\mathbb{i}_{\mathsf{GBF}}$ outputs short index $\iota = (n, d_{\nu}, t_M, \mathsf{vk}, \mathsf{vk}_2, \{[M_i(\mathbf{X},\mathbf{Y})]\}_{i=1}^{t_M})$ and index $\mathbb{i} = (\mathbb{i}_{\mathsf{PCE}}, \mathbb{i}_{\mathsf{hbPCE}}) = (\perp, (n, d_{\nu}, t_M, \{M_i(\mathbf{X},\mathbf{Y})\}_{i=1}^{t_M}))$.

Completeness. It is immediate from the perfect completeness of the sumcheck argument and the following equations.

	$\nu = 1$	$\nu = \log n$
Round complexity	3	$2 + \log n$
Prover communication		
Commitments	$2 + \sum_{k\in[K]} t_{M,v}^{(k)}$	$\sum_{k\in[K]} t_{M,v}^{(k)}$
Field elements	$n_{F,\nu}$	$n_{F,\nu}$
Polynomials	–	$\log n$ univariate, degree $d_l + d_r$
Verifier		
Randomness	$K + 2 + \sum_{k\in[K]} t_{M,v}^{(k)}$	$K + \log n + \sum_{k\in[K]} t_{M,v}^{(k)}$
Complexity	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$
Knowledge soundness		
Extractor complexity	$\mathcal{O}(n)$	$\mathcal{O}(n)$
Knowledge error	$\left(\frac{n+1}{\lvert\mathbb{F}\rvert}, \frac{(n-1)(d_l+d_r)}{\lvert\mathbb{F}\rvert}\right)$	$\left(\frac{\log n+2}{\lvert\mathbb{F}\rvert}, \frac{d_l+d_r}{\lvert\mathbb{F}\rvert}, \ldots, \frac{d_l+d_r}{\lvert\mathbb{F}\rvert}\right)$

Table 1. Efficiency and round-by-round knowledge soundness of Π_{GBF1}, in the univariate ($\nu = 1$) and multivariate ($\nu = \log n$) instantiations. Here $n_{F,\nu} = 2 + t_M + \sum_{k\in[K]} \left(t_u^{(k)} + t_v^{(k)} + t_{M,v}^{(k)}\right)$ if $\nu = 1$ and $n_{F,\nu} = t_M + \sum_{k\in[K]} \left(t_u^{(k)} + t_v^{(k)} + t_{M,v}^{(k)}\right)$ if $\nu = \log n$.

$$\sum_{\mathbf{h}\in\mathbb{H}} q_{\mathsf{IP}}^{(k)}(\mathbf{h}) = \left(\sum_{i=1}^{q_l^{(k)}} c_{l,i}^{(k)} \bigcirc_{j\in S_{l,i}^{(k)}} \mathbf{u}_j^{(k)}\right)^{\top} \left(\sum_{i=1}^{q_r^{(k)}} c_{r,i}^{(k)} \bigcirc_{(j_M,j_v)\in S_{r,i}^{(k)}} \mathbf{d}_{j_M,j_v}^{(k)}\right)$$

$$\sum_{\mathbf{h}\in\mathbb{H}} \left(d_{j_M,j_v}^{(k)}(\mathbf{h})\Lambda(\mathbf{h},\boldsymbol{\alpha}) - M_{j_M}(\mathbf{h},\boldsymbol{\alpha}) v_{j_v}^{(k)}(\mathbf{h})\right) = \mathbf{d}_{j_M,j_v}^{(k)\top}\boldsymbol{\lambda}(\boldsymbol{\alpha}) - \boldsymbol{\lambda}(\boldsymbol{\alpha})^{\top}\mathbf{M}_{j_M}\mathbf{v}_{j_v}^{(k)}$$

Next we analyze the round-by-round knowledge soundness.

Round-by-round knowledge soundness. *Notation for the analysis.* Let $u_i^{(k)}(\mathbf{X})$, $v_i^{(k)}(\mathbf{X})$, $d_{j_M,j_v}^{(k)}(\mathbf{X})$ be the polynomials extracted by $\mathcal{E}_{\mathsf{PC}}$, and define

$$q_{\mathsf{IP}}^{(k)}(\mathbf{X}) \coloneqq \left(\sum_{i=1}^{q_l^{(k)}} c_{l,i}^{(k)} \prod_{j\in S_{l,i}^{(k)}} u_j^{(k)}(\mathbf{X})\right)\left(\sum_{i=1}^{q_r^{(k)}} c_{r,i}^{(k)} \prod_{(j_M,j_v)\in S_{r,i}^{(k)}} d_{j_M,j_v}^{(k)}(\mathbf{X})\right),$$

$$q_{\mathsf{Lin}}^{(k)}(\mathbf{X}) \coloneqq \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} \left(d_{j_M,j_v}^{(k)}(\mathbf{X})\,\Lambda(\mathbf{X},\boldsymbol{\alpha}) - M_{j_M}(\mathbf{X},\boldsymbol{\alpha})\, v_{j_v}^{(k)}(\mathbf{X})\right),$$

and set $q(\mathbf{X}) \coloneqq \sum_{k\in[K]} \gamma^{(k)}\left(q_{\mathsf{IP}}^{(k)}(\mathbf{X}) + q_{\mathsf{Lin}}^{(k)}(\mathbf{X})\right)$ and $s \coloneqq \sum_{k\in[K]} \gamma^{(k)} s^{(k)}$. We write $\pi_d \coloneqq \{[d_{j_M,j_v}^{(k)}(\mathbf{X})]\}_{(j_M,j_v)\in S_R^{(k)}}^{k\in[K]}$ for the prover's first message and $\rho \coloneqq (\boldsymbol{\alpha}, \{\boldsymbol{\eta}^{(k)}\}^{k\in[K]}, \boldsymbol{\gamma})$ for the verifier's first challenge bundle. In the case $\nu = \log n$ we call a partial transcript **sumcheck-valid** if the verifier's checks (1)–(3) pass for all messages it contains; note that no extension of a non-sumcheck-valid transcript is sumcheck-valid.

State function and extractor.

(i) If $\mathsf{tr} = \emptyset$, $\mathsf{State}(\mathbb{i}, \mathbb{x}, \mathsf{tr}) = 1$ iff $(\mathbb{i}, \mathbb{x}) \in \mathcal{L}(\mathcal{R}^*_{\mathsf{GBF}})$. The extractor uses $\mathcal{E}_{\mathsf{PC}}$ to get $u_i^{(k)}(\mathbf{X}), v_i^{(k)}(\mathbf{X})$ and outputs $\mathbb{w} = \left(\{\mathbf{u}_i^{(k)}\}_{i\in[t_u^{(k)}]}^{k\in[K]}, \{\mathbf{v}_i\}_{i\in[t_v^{(k)}]}^{k\in[K]}\right)$ such that $u_{ij}^{(k)} = u_i^{(k)}(\mathbf{h}_j)$ and $v_{lj} = v_l^{(k)}(\mathbf{h}_j)$, $\forall i \in [t_u^{(k)}], l \in [t_v^{(k)}], j \in [n], k \in [K]$.

(ii) After $\pi_d \| \rho$, $\mathsf{State} = 1$ iff $\sum_{\mathbf{h}\in\mathbb{H}} q(\mathbf{h}) = s$.

(iii) In the sumcheck phase (with $h_1(X), h_2(X)$ extracted by $\mathcal{E}_{\mathsf{PC}}$):

- $\underline{\nu = 1:}$ after the prover sends $[h_1(X)], [h_2(X)]$ and $\mathcal{V}$ responds β, $\mathsf{State} = 1$ iff $q(\beta) = sn^{-1} + \beta h_1(\beta) + u_{\mathbb{H}}(\beta)h_2(\beta)$.
- $\underline{\nu = \log n}$: for $1 \le i \le \nu$, after the prover sends q_i and $\mathcal{V}$ responds β_i, $\mathsf{State} = 1$ iff tr is sumcheck-valid and $q_i(\beta_i) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q(\beta_1, \ldots, \beta_i, \mathbf{b})$.

Bounding the errors.

– $\underline{\text{Bounding } \epsilon_1}$: We show that if $\mathsf{State}(\pi_d \| \rho) = 1$ then, except with probability ϵ_1 over ρ, also $\mathsf{State}(\pi_d) = 1$. Assume $\mathsf{State}(\pi_d \| \rho) = 1$, i.e. $\sum_{\mathbf{h}\in\mathbb{H}} q(\mathbf{h}) = s$. Since $\boldsymbol{\gamma}$ was defined after the prover's message, except with probability $\frac{1}{|\mathbb{F}|}$ it holds that $\forall k \in [K]$, $\sum_{\mathbf{h}\in\mathbb{H}} \left(q_{\mathsf{IP}}^{(k)}(\mathbf{h}) + q_{\mathsf{Lin}}^{(k)}(\mathbf{h})\right) = s^{(k)}$. Since the $\boldsymbol{\eta}^{(k)}$ were defined after the prover's message, except with probability $\frac{1}{|\mathbb{F}|}$ this gives, $\forall k \in [K]$,

(1) $\sum_{\mathbf{h}\in\mathbb{H}} \left(\sum_{i=1}^{q_l^{(k)}} c_{l,i}^{(k)} \prod_{j\in S_{l,i}^{(k)}} u_j^{(k)}(\mathbf{h})\right)\left(\sum_{i=1}^{q_r^{(k)}} c_{r,i}^{(k)} \prod_{(j_M,j_v)\in S_{r,i}^{(k)}} d_{j_M,j_v}^{(k)}(\mathbf{h})\right) = s^{(k)}$, and

(2) $\forall (j_M, j_v) \in S_R^{(k)}$, $\sum_{\mathbf{h}\in\mathbb{H}} \left(d_{j_M,j_v}^{(k)}(\mathbf{h})\Lambda(\mathbf{h}, \boldsymbol{\alpha}) - M_{j_M}(\mathbf{h}, \boldsymbol{\alpha}) v_{j_v}^{(k)}(\mathbf{h})\right) = 0$.

Since $\boldsymbol{\alpha}$ was defined after the prover's message, except with probability $\frac{n-1}{|\mathbb{F}|}$ when $\nu = 1$ and $\frac{\log n}{|\mathbb{F}|}$ when $\nu = \log n$, (2) forces $d_{j_M,j_v}^{(k)}(\mathbf{h}) = \sum_{\mathbf{h}'\in\mathbb{H}} M_{j_M}(\mathbf{h}', \mathbf{h}) v_{j_v}^{(k)}(\mathbf{h}')$ for all $\mathbf{h} \in \mathbb{H}$, whence $\mathsf{State}(\pi_d) = 1$. By a union bound $\epsilon_1 \le \frac{n+1}{|\mathbb{F}|}$ when $\nu = 1$ and $\epsilon_1 \le \frac{\log n + 2}{|\mathbb{F}|}$ when $\nu = \log n$.

– $\underline{\text{Bounding } \epsilon_2}$:

- $\underline{\nu = 1:}$ with tr ending at $[h_1(X)], [h_2(X)]$ and $\mathcal{V}$ responding β, suppose $\mathsf{State}(\mathsf{tr}) = 0$ and $\mathsf{State}(\mathsf{tr} \| \beta) = 1$. Since $\mathsf{State}(\mathsf{tr}) = 0$, $\sum_{h\in\mathbb{H}} q(h) = s' \ne s$, hence $q(X) \ne sn^{-1} + Xh_1(X) + u_{\mathbb{H}}(X)h_2(X)$ for any $h_1(X), h_2(X)$ of appropriate degree. Since $\mathsf{State}(\mathsf{tr} \| \beta) = 1$, it must hold that $q(\beta) = sn^{-1} + \beta h_1(\beta) + u_{\mathbb{H}}(\beta)h_2(\beta)$, which can happen only with probability $\epsilon_2 = \frac{(n-1)(d_l+d_r)}{|\mathbb{F}|}$.
- $\underline{\nu = \log n, \text{ iteration } i}$: with tr ending at $\mathsf{tr}_{\mathsf{SC}}^i = (q_1, \beta_1, \ldots, \beta_{i-1}, q_i)$ and $\mathcal{V}$ responding β_i, suppose $\mathsf{State}(\mathsf{tr}) = 0$ and $\mathsf{State}(\mathsf{tr} \| \beta_i) = 1$; then tr is sumcheck-valid. If $q_i(X) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q(\beta_1, \ldots, \beta_{i-1}, X, \mathbf{b})$, then since tr is sumcheck-valid, $q_{i-1}(\beta_{i-1}) = q_i(0) + q_i(1) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-(i-1)}} q(\beta_1, \ldots, \beta_{i-1}, \mathbf{b})$, contradicting $\mathsf{State}(\mathsf{tr}) = 0$. Hence $q_i(X) \ne \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q(\beta_1, \ldots, \beta_{i-1}, X, \mathbf{b})$, and they agree at the random β_i with probability at most $\epsilon_2 = \frac{d_l+d_r}{|\mathbb{F}|}$.

Finally, we show State is consistent with the verifier's decision. Let tr contain all but the prover's final message Evals, with $\mathsf{State}(\mathsf{tr}) = 0$, and suppose $\mathcal{V}$ accepts; let ζ be as in the decision phase. Then:

– $\underline{\nu = 1:}$ $\zeta = sn^{-1} + \beta h_1 + u_{\mathbb{H}}(\beta)h_2 \ne q(\beta)$.

– $\underline{\nu = \log n}$: tr is sumcheck-valid and $q_\nu(\beta_\nu) \ne q(\boldsymbol{\beta})$, hence $\zeta = q_\nu(\beta_\nu) \ne q(\boldsymbol{\beta})$.

We conclude that this can happen only if at least one of the evaluations is wrong. □

(a) $\mathcal{P}$ computes $d^{(k)}_{j_M,j_v}(\mathbf{X}) := \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{M}_{j_M} \mathbf{v}^{(k)}_{j_v}$, and $[d^{(k)}_{j_M,j_v}(\mathbf{X})]$, $\forall (j_M, j_v) \in S^{(k)}_R$.

$$\longrightarrow \quad \{[d^{(k)}_{j_M,j_v}(\mathbf{X})]\}^{k\in[K]}_{(j_M,j_v)\in S^{(k)}_R} \quad \longrightarrow$$

$$\longleftarrow \quad \boldsymbol{\alpha} \in \mathbb{F}^\nu, \{\boldsymbol{\eta}^{(k)} \in \mathbb{F}^{t^{(k)}_{M,v}}\}^{k\in[K]}, \boldsymbol{\gamma} \in \mathbb{F}^K \quad \longleftarrow$$

(b) $\mathcal{P}$ computes $q(\mathbf{X}) = \sum_{k\in[K]} \gamma^{(k)} (q^{(k)}_{\mathsf{IP}}(\mathbf{X}) + q^{(k)}_{\mathsf{Lin}}(\mathbf{X}))$, where

$$q^{(k)}_{\mathsf{IP}}(\mathbf{X}) := \left(\sum_{i=1}^{q^{(k)}_l} c^{(k)}_{l,i} \prod_{j\in S^{(k)}_{l,i}} u^{(k)}_j(\mathbf{X}) \right) \left(\sum_{i=1}^{q^{(k)}_r} c^{(k)}_{r,i} \prod_{(j_M,j_v)\in S^{(k)}_{r,i}} d^{(k)}_{j_M,j_v}(\mathbf{X}) \right)$$

and

$$q^{(k)}_{\mathsf{Lin}}(\mathbf{X}) := \sum_{(j_M,j_v)\in S^{(k)}_R} \eta^{(k)}_{j_M,j_v} \left(d^{(k)}_{j_M,j_v}(\mathbf{X}) \Lambda(\mathbf{X}, \boldsymbol{\alpha}) - M_{j_M}(\mathbf{X}, \boldsymbol{\alpha}) v^{(k)}_{j_v}(\mathbf{X}) \right)$$

It also computes $s = \sum_{k\in[K]} \gamma^{(k)} s^{(k)}$, and interacts with $\mathcal{V}$ as follows:

- $\underline{\nu = 1}$: $\mathcal{P}$ computes polynomials $h_1(X) \in \mathbb{F}[X]^{\leq n-2}$, $h_2(X) \in \mathbb{F}[X]^{\leq n(d_l+d_r-1)-(d_l+d_r)}$ and $[h_1(X)], [h_2(X)]$, such that $q(X) = sn^{-1} + Xh_1(X) + u_{\mathbb{H}}(X)h_2(X)$.

$$\longrightarrow \quad [h_1(X)], [h_2(X)] \quad \longrightarrow$$

$$\longleftarrow \quad \beta \in \mathbb{F} \quad \longleftarrow$$

- $\underline{\nu = \log n}$: Initialize $\boldsymbol{\beta} = \emptyset$. For $1 \leq i \leq \nu$ do the following:
  $\mathcal{P}$ computes $q_i(X) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q(\boldsymbol{\beta}, X, \mathbf{b})$.

$$\longrightarrow \quad q_i(X) \quad \longrightarrow$$

$$\longleftarrow \quad \beta_i \in \mathbb{F} \quad \longleftarrow$$

  Append β_i to $\boldsymbol{\beta}$.

(c) $\mathcal{P}$ computes the evaluation set Evals which includes $\{u^{(k)}_i(\boldsymbol{\beta})\}^{k\in[K]}_{i\in[t_u]}$, $\{v^{(k)}_i(\boldsymbol{\beta})\}^{k\in[K]}_{i\in[t_v]}$, $\{d^{(k)}_{j_M,j_v}(\boldsymbol{\beta})\}^{k\in[K]}_{(j_M,j_v)\in S_R}$, $\{M_i(\boldsymbol{\beta}, \boldsymbol{\alpha})\}_{i\in[t_M]}$. In case $\nu = 1$, $\mathcal{P}$ also includes $h_1(\beta), h_2(\beta)$.

$$\longrightarrow \quad \mathsf{Evals} \quad \longrightarrow$$

Fig. 2. Interaction phase of Π_{GBF1}.

$\mathcal{V}$ parses Evals as $\{\{u_i^{(k)}\}_{i\in[t_u]}^{k\in[K]}, \{v_i^{(k)}\}_{i\in[t_v]}^{k\in[K]}, \{d_{j_M,j_v}^{(k)}\}_{(j_M,j_v)\in S_R}^{k\in[K]}, \{m_i\}_{i\in[t_M]}, h_1, h_2\}$, computes

$$\zeta = \sum_{k=1}^{K} \gamma^{(k)} \left(\zeta_1^{(k)} + \zeta_2^{(k)}\right),$$

where

$$\zeta_1^{(k)} := \left(\sum_{i=1}^{q_l^{(k)}} c_{l,i}^{(k)} \prod_{j\in S_{l,i}^{(k)}} u_j^{(k)}\right)\left(\sum_{i=1}^{q_r^{(k)}} c_{r,i}^{(k)} \prod_{(j_M,j_v)\in S_{r,i}^{(k)}} d_{j_M,j_v}^{(k)}\right),$$

and

$$\zeta_2^{(k)} := \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} \left(d_{j_M,j_v}^{(k)} \Lambda(\boldsymbol{\beta}, \boldsymbol{\alpha}) - m_{j_M} v_{j_v}^{(k)}\right).$$

It also computes $s = \sum_{k\in[K]} \gamma^{(k)} s^{(k)}$.

Decision. $\mathcal{V}$ checks

- $\nu = 1$: $sn^{-1} + \beta h_1 + u_{\mathbb{H}}(\beta) h_2 = \zeta$
- $\nu = \log n$: (1) $\forall i \in [\nu], q_i(X) \in \mathbb{F}[X]^{\leq d_l + d_r}$, (2) $q_1(0) + q_1(1) = s$, (3) $\forall i > 1$ $q_i(\beta_i) = q_{i+1}(0) + q_{i+1}(1)$, and (4) $q_\nu(\beta_\nu) = \zeta$.

Output. $\mathcal{P}$ and $\mathcal{V}$ output

(1) $\mathcal{R}_{\mathsf{PCE}}$ instances $\mathbb{x}_{\mathsf{PCE}}^{(g)} = (d_g, [g(\mathbf{X})], \boldsymbol{\beta}, g)$, where $g \in \{\{u_i^{(k)}\}_{i\in[t_u]}^{k\in[K]}, \{v_i^{(k)}\}_{i\in[t_v]}^{k\in[K]}, \{d_{j_M,j_v}^{(k)}\}_{(j_M,j_v)\in S_R^{(k)}}^{k\in[K]}, h_1, h_2\}$, d_g as defined in the protocol, and
(2) $\mathcal{R}_{\mathsf{hbPCE}}$ instance $\mathbb{x}_{\mathsf{hbPCE}} = (\{[M_i(\mathbf{X}, \mathbf{Y})], m_i\}_{i=1}^{t_M}, \boldsymbol{\alpha}, \boldsymbol{\beta})$.

$\mathcal{P}$ also outputs $\mathbb{w}_{\mathsf{PCE}}^{(g)} = g(\mathbf{X})$, where $g \in \{\{u_i^{(k)}\}_{i\in[t_u]}^{k\in[K]}, \{v_i^{(k)}\}_{i\in[t_v]}^{k\in[K]}, \{d_{j_M,j_v}^{(k)}\}_{(j_M,j_v)\in S_R^{(k)}}^{k\in[K]}, h_1, h_2\}$.

Fig. 3. Decision and output phase of Π_{GBF1}.

	$\nu = 1$	$\nu = \log n$
Round complexity	4	$4 + 2\log n$
Prover communication		
Commitments	4	–
Field elements	$n_{F,\nu}$	$n_{F,\nu}$
Polynomials (deg. $d_l + d_r$)	–	$\log n$ univariate
Polynomials (deg. 2)	–	$\log n$ univariate
Verifier		
Randomness	$K + 2 + \sum_{k\in[K]} t_{M,v}^{(k)}$	$K + 2\log n + \sum_{k\in[K]} t_{M,v}^{(k)}$
Complexity	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$
Knowledge soundness		
Extractor complexity	$\mathcal{O}(n)$	$\mathcal{O}(n)$
Knowledge error	$\left(\frac{1}{\lvert\mathbb{F}\rvert}, \frac{(n-1)(d_l+d_r)}{\lvert\mathbb{F}\rvert}, \frac{1}{\lvert\mathbb{F}\rvert}, \frac{2n-2}{\lvert\mathbb{F}\rvert}\right)$	$\left(\frac{1}{\lvert\mathbb{F}\rvert}, \frac{d_l+d_r}{\lvert\mathbb{F}\rvert}, \ldots, \frac{d_l+d_r}{\lvert\mathbb{F}\rvert}, \frac{1}{\lvert\mathbb{F}\rvert}, \frac{2}{\lvert\mathbb{F}\rvert}, \ldots, \frac{2}{\lvert\mathbb{F}\rvert}\right)$

Table 2. Efficiency and round-by-round knowledge soundness of Π_{GBF2}, in the univariate ($\nu = 1$) and multivariate ($\nu = \log n$) instantiations. Here $n_{F,\nu} = 4 + t_M + \sum_{k\in[K]}\left(t_u^{(k)} + t_v^{(k)} + t_{M,v}^{(k)}\right)$ if $\nu = 1$ and $n_{F,\nu} = t_M + \sum_{k\in[K]}\left(t_u^{(k)} + t_v^{(k)} + t_{M,v}^{(k)}\right)$ if $\nu = \log n$.

4.2 Π_{GBF2}

In this section we present the interactive reduction of knowledge Π_{GBF2}. The protocol uses two invocations of the sumcheck argument, similar to the Spartan [Set20] approach for proving R1CS satisfiability.

Theorem 6. *The protocol Π_{GBF2} of Fig. 4, 5 is an interactive reduction of knowledge from $\mathcal{R}^*_{\mathsf{GBF}}$ to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}}$ with perfect completeness. with perfect completeness. Its efficiency and round-by-round knowledge-soundness properties are summarized in Table 2.*

Proof. The indexer $\mathcal{I}$ on input public parameters pp and index $\mathbb{i}_{\mathsf{GBF}}$ outputs short index $\iota = (n, d_\nu, t_M, \{[M_i(\mathbf{X},\mathbf{Y})]\}_{i=1}^{t_M})$ and index $\mathbb{i} = (\mathbb{i}_{\mathsf{PCE}}, \mathbb{i}_{\mathsf{hbPCE}}) = (\perp, (n, d_\nu, t_M, \{M_i(\mathbf{X},\mathbf{Y})\}_{i=1}^{t_M}))$.

Completeness. It is immediate from the perfect completeness of the sumcheck argument and the following equations.

$$\sum_{\mathbf{h}\in\mathbb{H}} q^{(k)}(\mathbf{h}) = \left(\sum_{i=1}^{q_l^{(k)}} c_{l,i}^{(k)} \bigcirc_{j\in S_{l,i}^{(k)}} \mathbf{u}_j^{(k)}\right)^\top \left(\sum_{i=1}^{q_r^{(k)}} c_{r,i}^{(k)} \bigcirc_{(j_M,j_v)\in S_{r,i}^{(k)}} \mathbf{M}_{j_M}\mathbf{v}_{j_v}^{(k)}\right)$$

$$\sum_{\mathbf{h}\in\mathbb{H}} \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M}\boldsymbol{\lambda}(\mathbf{h}) \cdot \mathbf{v}_{j_v}^{(k)\top}\boldsymbol{\lambda}(\mathbf{h}) = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M}\mathbf{v}_{j_v}^{(k)} = \zeta_{j_M,j_v}^{(k)}$$

Next we analyze the round-by-round knowledge soundness.

Round-by-round knowledge soundness. *Notation for the analysis.* Let $u_i^{(k)}(\mathbf{X})$, $v_i^{(k)}(\mathbf{X})$ be extracted by $\mathcal{E}_{\mathsf{PC}}$, and define

$$q^{(k)}(\mathbf{X}) := \left(\sum_{i=1}^{q_l^{(k)}} c_{l,i}^{(k)} \prod_{j\in S_{l,i}^{(k)}} u_j^{(k)}(\mathbf{X})\right)\left(\sum_{i=1}^{q_r^{(k)}} c_{r,i}^{(k)} \prod_{(j_M,j_v)\in S_{r,i}^{(k)}} \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{M}_{j_M}\mathbf{v}_{j_v}^{(k)}\right),$$

$$q'(\mathbf{X}) := \sum_{k\in[K]} \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)}\, \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M} \boldsymbol{\lambda}(\mathbf{X})\, v_{j_v}^{(k)}(\mathbf{X}),$$

and set $q := \sum_{k\in[K]} \gamma^{(k)} q^{(k)}$, $s := \sum_{k\in[K]} \gamma^{(k)} s^{(k)}$, and $s' := \sum_{k\in[K]} \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} \zeta_{j_M,j_v}^{(k)}$, where ζ, ζ' are the values computed in the decision phase of $\mathcal{V}$. For $\nu = \log n$, a partial transcript is **sumcheck-valid** as in Π_{GBF1} (applied to whichever sumcheck its messages belong to).

State function and extractor. The protocol proceeds in the order: challenge $\boldsymbol{\gamma}$; first sumcheck (challenge α, resp. $\alpha_1, \ldots, \alpha_\nu$); challenge $\{\boldsymbol{\eta}^{(k)}\}^{k\in[K]}$; second sumcheck (challenge β, resp. $\beta_1, \ldots, \beta_\nu$); Evals.

(i) If $\mathsf{tr} = \emptyset$, $\mathsf{State} = 1$ iff $(\mathbb{i}, \mathbb{x}) \in \mathcal{L}(\mathcal{R}^*_{\mathsf{GBF}})$. The extractor uses $\mathcal{E}_{\mathsf{PC}}$ to get $u_i^{(k)}(\mathbf{X}), v_i^{(k)}(\mathbf{X})$ and outputs $\mathbb{w} = \left(\{\mathbf{u}_i^{(k)}\}^{k\in[K]}_{i\in[t_u^{(k)}]}, \{\mathbf{v}_i\}^{k\in[K]}_{i\in[t_v^{(k)}]}\right)$ such that $u_{ij}^{(k)} = u_i^{(k)}(\mathbf{h}_j)$ and $v_{lj} = v_l^{(k)}(\mathbf{h}_j)$, $\forall i \in [t_u^{(k)}], l \in [t_v^{(k)}], j \in [n], k \in [K]$.

(ii) After $\boldsymbol{\gamma}$, $\mathsf{State} = 1$ iff $\sum_{\mathbf{h}\in\mathbb{H}} q(\mathbf{h}) = s$.

(iii) First sumcheck:
- $\underline{\nu = 1:}$ after $[h_1(X)], [h_2(X)]$ and challenge α, $\mathsf{State} = 1$ iff $q(\alpha) = sn^{-1} + \alpha h_1(\alpha) + u_{\mathbb{H}}(\alpha) h_2(\alpha)$.
- $\underline{\nu = \log n}$: for $1 \le i \le \nu$, after q_i and challenge α_i, $\mathsf{State} = 1$ iff tr is sumcheck-valid and $q_i(\alpha_i) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q(\alpha_1, \ldots, \alpha_i, \mathbf{b})$.

(iv) After $\{\boldsymbol{\eta}^{(k)}\}^{k\in[K]}$:
- $\underline{\nu = 1:}$ $\mathsf{State} = 1$ iff $\zeta = sn^{-1} + \alpha h_1 + u_{\mathbb{H}}(\alpha) h_2$ and $\sum_{h\in\mathbb{H}} q'(h) = s'$.
- $\underline{\nu = \log n}$: $\mathsf{State} = 1$ iff tr is sumcheck-valid, $\zeta = q_\nu(\alpha_\nu)$, and $\sum_{\mathbf{b}\in\{0,1\}^\nu} q'(\mathbf{b}) = s'$.

(v) Second sumcheck:
- $\underline{\nu = 1:}$ after $[h_1'(X)], [h_2'(X)]$ and challenge β, $\mathsf{State} = 1$ iff $\zeta = sn^{-1} + \alpha h_1 + u_{\mathbb{H}}(\alpha) h_2$ and $q'(\beta) = s'n^{-1} + \beta h_1'(\beta) + u_{\mathbb{H}}(\beta) h_2'(\beta)$.
- $\underline{\nu = \log n}$: for $1 \le i \le \nu$, after q_i' and challenge β_i, $\mathsf{State} = 1$ iff tr is sumcheck-valid, $\zeta = q_\nu(\alpha_\nu)$, and $q_i'(\beta_i) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q'(\beta_1, \ldots, \beta_i, \mathbf{b})$.

Bounding the errors.

– $\underline{\text{Bounding } \epsilon_1}$: If $\mathsf{State}(\emptyset \| \boldsymbol{\gamma}) = 1$, i.e. $\sum_{\mathbf{h}\in\mathbb{H}} q(\mathbf{h}) = s$, then since $\boldsymbol{\gamma}$ was defined after the prover's message, except with probability $\epsilon_1 = \frac{1}{|\mathbb{F}|}$ it holds that $\forall k \in [K], \sum_{\mathbf{h}\in\mathbb{H}} q^{(k)}(\mathbf{h}) = s^{(k)}$, whence $(\mathbb{i}, \mathbb{x}^{(k)}) \in \mathcal{L}(\mathcal{R}_{\mathsf{GBF}})$ and $\mathsf{State}(\emptyset) = 1$.

– $\underline{\text{Bounding } \epsilon_2}$:
- $\underline{\nu = 1:}$ with tr ending at $[h_1(X)], [h_2(X)]$ and $\mathcal{V}$ responding α, suppose $\mathsf{State}(\mathsf{tr}) = 0$ and $\mathsf{State}(\mathsf{tr} \| \alpha) = 1$. Since $\mathsf{State}(\mathsf{tr}) = 0$, $\sum_{h\in\mathbb{H}} q(h) = s'' \neq s$, thus $q(X) \neq sn^{-1} + Xh_1(X) + u_{\mathbb{H}}(X) h_2(X)$ for any $h_1(X), h_2(X)$ of appropriate degree. Since $\mathsf{State}(\mathsf{tr} \| \alpha) = 1$, it must hold that $q(\alpha) = sn^{-1} + \alpha h_1(\alpha) + u_{\mathbb{H}}(\alpha) h_2(\alpha)$, which can happen only with probability $\epsilon_2 = \frac{(n-1)(d_l+d_r)}{|\mathbb{F}|}$.
- $\underline{\nu = \log n, \text{ iteration } i:}$ with tr ending at $(q_1, \alpha_1, \ldots, \alpha_{i-1}, q_i)$ and $\mathcal{V}$ responding α_i, suppose $\mathsf{State}(\mathsf{tr}) = 0$ and $\mathsf{State}(\mathsf{tr} \| \alpha_i) = 1$; then tr is sumcheck-valid. If $q_i(X) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q(\alpha_1, \ldots, \alpha_{i-1}, X, \mathbf{b})$, then since tr is sumcheck-valid, $q_{i-1}(\alpha_{i-1}) = q_i(0) + q_i(1) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-(i-1)}} q(\alpha_1, \ldots, \alpha_{i-1}, \mathbf{b})$, contradicting $\mathsf{State}(\mathsf{tr}) = 0$. Hence $q_i(X) \neq \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q(\alpha_1, \ldots, \alpha_{i-1}, X, \mathbf{b})$, and they agree at the random α_i with probability at most $\epsilon_2 = \frac{d_l+d_r}{|\mathbb{F}|}$.

– $\underline{\text{Bounding } \epsilon_3}$:
- $\underline{\nu = 1:}$ with tr ending at $\alpha, \{\mathbf{u}^{(k)}\}^{k\in[K]}, \{\boldsymbol{\zeta}^{(k)}\}^{k\in[K]}, h_1, h_2$ and $\mathcal{V}$ responding $\{\boldsymbol{\eta}^{(k)}\}^{k\in[K]}$, suppose $\mathsf{State}(\mathsf{tr}) = 0$ and $\mathsf{State}(\mathsf{tr} \| \{\boldsymbol{\eta}^{(k)}\}) = 1$. Since $\mathsf{State}(\mathsf{tr}) = 0$, $q(\alpha) \neq sn^{-1} + \alpha h_1(\alpha) + u_{\mathbb{H}}(\alpha) h_2(\alpha)$, while $\mathsf{State}(\mathsf{tr} \| \{\boldsymbol{\eta}^{(k)}\}) = 1$ forces $\zeta \neq q(\alpha)$. Hence either $\mathcal{V}$ accepts an invalid

evaluation claim for some $u_i^{(k)}(X)$, or for some $(j_M, j_v) \in S_R^{(k)}$, $\zeta_{j_M,j_v}^{(k)} \neq \boldsymbol{\lambda}(\alpha)^\top \mathbf{M}_{j_M} \mathbf{v}_{j_v}^{(k)} = \sum_{h\in\mathbb{H}} \boldsymbol{\lambda}(\alpha)^\top \mathbf{M}_{j_M} \boldsymbol{\lambda}(h)\, \mathbf{v}_{j_v}^{(k)\top} \boldsymbol{\lambda}(h)$. Since

$$\sum_{k\in[K]} \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} \Big(\sum_{h\in\mathbb{H}} \boldsymbol{\lambda}(\alpha)^\top \mathbf{M}_{j_M} \boldsymbol{\lambda}(h)\, \mathbf{v}_{j_v}^{(k)\top} \boldsymbol{\lambda}(h) \Big)$$
$$= \sum_{h\in\mathbb{H}} q'(h) = s' = \sum_{k\in[K]} \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} \zeta_{j_M,j_v}^{(k)},$$

this can happen with probability at most $\epsilon_3 = \frac{1}{|\mathbb{F}|}$ over the choice of $\boldsymbol{\eta}$.

- $\nu = \log n$: with $\mathsf{tr} = (\gamma, \mathsf{tr}_{\mathsf{SC1}}, \{\mathbf{u}^{(k)}\}^{k\in[K]}, \{\boldsymbol{\zeta}^{(k)}\}^{k\in[K]})$, $\mathsf{tr}_{\mathsf{SC1}} = (q_1, \alpha_1, \ldots, q_\nu, \alpha_\nu)$, and $\mathcal{V}$ responding $\{\boldsymbol{\eta}^{(k)}\}^{k\in[K]}$, suppose $\mathsf{State}(\mathsf{tr}) = 0$ and $\mathsf{State}(\mathsf{tr}\|\{\boldsymbol{\eta}^{(k)}\}) = 1$; then tr is sumcheck-valid. As in the case $\nu = 1$, either $\mathcal{V}$ accepts an invalid evaluation claim for some $u_i^{(k)}(X)$, or for some $(j_M, j_v) \in S_R^{(k)}$, $\zeta_{j_M,j_v}^{(k)} \neq \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M} \mathbf{v}_{j_v}^{(k)} = \sum_{\mathbf{h}\in\mathbb{H}} \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M} \boldsymbol{\lambda}(\mathbf{h})\, \mathbf{v}_{j_v}^{(k)\top} \boldsymbol{\lambda}(\mathbf{h})$; this happens with probability at most $\epsilon_3 = \frac{1}{|\mathbb{F}|}$ over $\boldsymbol{\eta}$.

– Bounding ϵ_4:
  - $\nu = 1$: with tr ending at $[h_1'(X)], [h_2'(X)]$ and $\mathcal{V}$ responding β, suppose $\mathsf{State}(\mathsf{tr}) = 0$ and $\mathsf{State}(\mathsf{tr}\|\beta) = 1$. Then $\zeta = sn^{-1} + \alpha h_1 + u_{\mathbb{H}}(\alpha) h_2$ (or else $\mathsf{State}(\mathsf{tr}\|\beta) = 0$), so we may assume $\sum_{h\in\mathbb{H}} q'(h) = s''' \neq s'$, meaning $q'(X) \neq s'n^{-1} + Xh_1'(X) + u_{\mathbb{H}}(X) h_2'(X)$ for any $h_1'(X), h_2'(X)$ of appropriate degree. Hence they agree at the random β with probability at most $\epsilon_4 = \frac{2n-2}{|\mathbb{F}|}$.
  - $\nu = \log n$, iteration i: with tr ending at $\mathsf{tr}_{\mathsf{SC}}^i = (q_1', \beta_1, \ldots, \beta_{i-1}, q_i')$ and $\mathcal{V}$ responding β_i, suppose $\mathsf{State}(\mathsf{tr}) = 0$ and $\mathsf{State}(\mathsf{tr}\|\beta_i) = 1$; then tr is sumcheck-valid and $\zeta = q_\nu(\alpha_\nu)$. If $q_i'(X) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q'(\beta_1, \ldots, \beta_{i-1}, X, \mathbf{b})$, then since tr is sumcheck-valid, $q_{i-1}'(\beta_{i-1}) = q_i'(0) + q_i'(1) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-(i-1)}} q'(\beta_1, \ldots, \beta_{i-1}, \mathbf{b})$, contradicting $\mathsf{State}(\mathsf{tr}) = 0$. Hence $q_i'(X) \neq \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q'(\beta_1, \ldots, \beta_{i-1}, X, \mathbf{b})$, and they agree at the random β_i with probability at most $\epsilon_4 = \frac{2}{|\mathbb{F}|}$.

Finally, we show State is consistent with the verifier's decision. Let tr contain all but the prover's final message Evals, with $\mathsf{State}(\mathsf{tr}) = 0$, and suppose $\mathcal{V}$ accepts; let ζ' be as in the decision phase. Then:

– $\nu = 1$: $\zeta' = s'n^{-1} + \beta h_1' + u_{\mathbb{H}}(\beta) h_2' \neq q'(\beta)$.
– $\nu = \log n$: tr is sumcheck-valid and $q_\nu'(\beta_\nu) \neq q'(\boldsymbol{\beta})$, hence $\zeta' = q_\nu'(\beta_\nu) \neq q'(\boldsymbol{\beta})$.

We conclude that this can happen only if at least one of the evaluations is wrong. □

4.3 Specializations

In this section we state a couple of useful corollaries that specialize Π_{GBF} to the cases $\mathcal{R}_{\mathsf{GBF},\alpha}$ and $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$.

Corollary 1. *There exists a family of interactive reductions of knowledge $\Pi_{\mathsf{GBF},\alpha}$ from $\mathcal{R}_{\mathsf{GBF},\alpha}^*$ to $\mathcal{R}_{\mathsf{PCE}}^* \times \mathcal{R}_{\mathsf{hbPCE}}$.*

Proof. We define the reductions $\Pi_{\mathsf{GBF1},\alpha}$ and $\Pi_{\mathsf{GBF2},\alpha}$ to work exactly as the reductions Π_{GBF1} and Π_{GBF2}, respectively, with the only difference that now $\mathcal{P}$ does not need to provide the evaluation of $u_j^{(i)}(\mathbf{X}) = \Lambda(\mathbf{X}, \boldsymbol{\alpha}^{(i)})$, since this can be computed by $\mathcal{V}$ in logarithmic time. □

Corollary 2. *There exists a family of interactive reductions of knowledge $\Pi_{\mathsf{GBF},\alpha,\beta}$ from $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}^*$ to $\mathcal{R}_{\mathsf{PCE}}^* \times \mathcal{R}_{\mathsf{hbPCE}}$.*

$\longleftarrow \quad \gamma \in \mathbb{F}^K \quad$

(a) $\mathcal{P}$ computes $s := \sum_{k\in[K]} \gamma^{(k)} s^{(k)}$ and $q(\mathbf{X}) := \sum_{k\in[K]} \gamma^{(k)} q^{(k)}(\mathbf{X})$, where $q^{(k)}(\mathbf{X}) := \left(\sum_{i=1}^{q_l^{(k)}} c_{l,i}^{(k)} \prod_{j\in S_{l,i}^{(k)}} u_j^{(k)}(\mathbf{X})\right)\left(\sum_{i=1}^{q_r^{(k)}} c_{r,i}^{(k)} \prod_{(j_M,j_v)\in S_{r,i}^{(k)}} \boldsymbol{\lambda}(\mathbf{X})^\top \mathbf{M}_{j_M} \mathbf{v}_{j_v}^{(k)}\right)$. It interacts with $\mathcal{V}$ as follows:

- $\underline{\nu = 1}$: $\mathcal{P}$ computes polynomials $h_1(X) \in \mathbb{F}[X]^{\leq n-2}$, $h_2(X) \in \mathbb{F}[X]^{\leq n(d_l+d_r-1)-(d_l+d_r)}$ and $[h_1(X)], [h_2(X)]$, such that $q(X) = sn^{-1} + Xh_1(X) + u_{\mathbb{H}}(X)h_2(X)$.

$\longrightarrow \quad [h_1(X)], [h_2(X)] \quad \longrightarrow$

$\longleftarrow \quad \alpha \in \mathbb{F} \quad$

- $\underline{\nu = \log n}$: Initialize $\boldsymbol{\alpha} = \emptyset$. For $1 \leq i \leq \nu$ do the following:
  $\mathcal{P}$ computes $q_i(X) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q(\boldsymbol{\alpha}, X, \mathbf{b})$.

$\longrightarrow \quad q_i(X) \quad \longrightarrow$

$\longleftarrow \quad \alpha_i \in \mathbb{F} \quad$

  Append α_i to $\boldsymbol{\alpha}$.

(b) $\mathcal{P}$ computes $\forall k \in [K]$ values $u_i^{(k)} = u_i^{(k)}(\boldsymbol{\alpha})\ \forall i \in [t_u]$, $\zeta_{j_M,j_v}^{(k)} = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M} \mathbf{v}_{j_v}^{(k)}$, $\forall (j_M, j_v) \in S_R^{(k)}$. In case $\nu = 1$ it also computes $h_1 = h_1(\alpha), h_2 = h_2(\alpha)$

$\longrightarrow \quad \{u_i^{(k)}\}_{i\in[t_u^{(k)}]}^{k\in[K]}, \{\zeta_{j_M,j_v}^{(k)}\}_{(j_M,j_v)\in S_R^{(k)}}^{k\in[K]}, h_1, h_2 \quad \longrightarrow$

$\longleftarrow \quad \{\boldsymbol{\eta}^{(k)} \in \mathbb{F}^{t_{M,v}^{(k)}}\}^{k\in[K]} \quad$

(c) $\mathcal{P}$ computes $s' = \sum_{k\in[K]} \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} \zeta_{j_M,j_v}^{(k)}$ and $q'(\mathbf{X}) = \sum_{k\in[K]} \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M} \boldsymbol{\lambda}(\mathbf{X}) v_{j_v}^{(k)}(\mathbf{X})$ and interacts with $\mathcal{V}$ as follows:

- $\underline{\nu = 1}$: $\mathcal{P}$ computes polynomials $h_1'(X), h_2'(X) \in \mathbb{F}[X]^{\leq n-2}$ and $[h_1'(X)], [h_2'(X)]$, such that $q'(X) = s'n^{-1} + Xh_1'(X) + u_{\mathbb{H}}(X)h_2'(X)$.

$\longrightarrow \quad [h_1'(X)], [h_2'(X)] \quad \longrightarrow$

$\longleftarrow \quad \beta \in \mathbb{F} \quad$

- $\underline{\nu = \log n}$: Initialize $\boldsymbol{\beta} = \emptyset$. For $1 \leq i \leq \nu$ do the following:
  $\mathcal{P}$ computes $q_i'(X) = \sum_{\mathbf{b}\in\{0,1\}^{\nu-i}} q'(\boldsymbol{\beta}, X, \mathbf{b})$.

$\longrightarrow \quad q_i'(X) \quad \longrightarrow$

$\longleftarrow \quad \beta_i \in \mathbb{F} \quad$

  Append β_i to $\boldsymbol{\beta}$.

(d) $\mathcal{P}$ computes the evaluation set $\mathsf{Evals} = \left\{\{v_i^{(k)}(\boldsymbol{\beta})\}_{i\in[t_u^{(k)}]}^{k\in[K]}, \{M_i(\boldsymbol{\beta}, \boldsymbol{\alpha})\}_{i\in[t_M]}^{k\in[K]}\right\}$. In case $\nu = 1$, $\mathcal{P}$ also includes $h_1'(\beta), h_2'(\beta)$.

$\longrightarrow \quad \mathsf{Evals} \quad \longrightarrow$

Fig. 4. Interaction phase of Π_{GBF2}.

$\mathcal{V}$ parses Evals as $\{\{v_i^{(k)}\}_{i\in[t_v]}^{k\in[K]}, \{m_i\}_{i\in[t_M]}, h_1', h_2'\}$, computes

$$s = \sum_{k\in[K]} \gamma^{(k)} s^{(k)},$$

$$s' = \sum_{k\in[K]} \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} \zeta_{j_M,j_v}^{(k)},$$

$$\zeta = \sum_{k\in[K]} \gamma^{(k)} \left(\sum_{i=1}^{q_l^{(k)}} c_{l,i}^{(k)} \prod_{j\in S_{l,i}^{(k)}} u_j^{(k)} \right) \left(\sum_{i=1}^{q_r^{(k)}} c_{r,i}^{(k)} \prod_{(j_M,j_v)\in S_{r,i}^{(k)}} \zeta_{j_M,j_v}^{(k)} \right), \text{ and}$$

$$\zeta' = \sum_{k\in[K]} \sum_{(j_M,j_v)\in S_R^{(k)}} \eta_{j_M,j_v}^{(k)} m_{j_M} v_{j_v}^{(k)}.$$

Decision. $\mathcal{V}$ checks

- $\nu = 1$: $sn^{-1} + \alpha h_1 + u_{\mathbb{H}}(\alpha) h_2 = \zeta$ and $s'n^{-1} + \beta h_1' + u_{\mathbb{H}}(\beta) h_2' = \zeta'$
- $\nu = \log n$: (1) $\forall i \in [\nu], q_i(X) \in \mathbb{F}[X]^{\leq d_l + d_r}$, $q_i'(X) \in \mathbb{F}[X]^{\leq 2}$ (2) $q_1(0) + q_1(1) = s$, $q_1'(0) + q_1'(1) = s'$ (3) $\forall i > 1$ $q_i(\alpha_i) = q_{i+1}(0) + q_{i+1}(1)$, $q_i'(\beta_i) = q_{i+1}'(0) + q_{i+1}'(1)$, and (4) $q_\nu(\alpha_\nu) = \zeta$, $q_\nu'(\beta_\nu) = \zeta'$

Output. $\mathcal{P}$ and $\mathcal{V}$ output

(1) $\mathcal{R}_{\mathsf{PCE}}$ instances $\mathbb{x}_{\mathsf{PCE}}^{(g)} = (d_g, [g(\mathbf{X})], \boldsymbol{\beta}, g)$ where $g \in \{\{u_i^{(k)}\}_{i\in[t_u]}^{k\in[K]}, \{v_i^{(k)}\}_{i\in[t_v]}^{k\in[K]}, h_1, h_2, h_1', h_2'\}$, d_g as defined in the protocol, and
(2) $\mathcal{R}_{\mathsf{hbPCE}}$ instance $\mathbb{x}_{\mathsf{hbPCE}} = (\{[M_i(\mathbf{X}, \mathbf{Y})], m_i\}_{i\in[t_M]}, \boldsymbol{\alpha}, \boldsymbol{\beta})$.

$\mathcal{P}$ also outputs $\mathbb{w}_{\mathsf{PCE}}^{(g)} = g(\mathbf{X})$, where $g \in \{\{u_i^{(k)}\}_{i\in[t_u]}^{k\in[K]}, \{v_i^{(k)}\}_{i\in[t_v]}^{k\in[K]}, h_1, h_2, h_1', h_2'\}$.

Fig. 5. Decision and output phase of Π_{GBF2}.

Proof. We define the reductions $\Pi_{\mathsf{GBF1},\alpha,\beta}$ and $\Pi_{\mathsf{GBF2},\alpha,\beta}$ to work exactly as the reductions Π_{GBF1} and Π_{GBF2}, respectively, with the only difference that now $\mathcal{P}$ does not need to provide the evaluations of $u_j^{(i)}(\mathbf{X})$ and $v_j^{(i)}(\mathbf{X})$. This is because in the case of $\mathcal{R}^*_{\mathsf{GBF},\alpha,\beta}$, $u^{(i)}(\mathbf{X}) = \Lambda(\mathbf{X}, \boldsymbol{\alpha}^{(i)})$ and $v^{(i)}(\mathbf{X}) = \Lambda(\mathbf{X}, \boldsymbol{\beta}^{(i)})$, and hence $\mathcal{V}$ can compute evaluations of these on its own in logarithmic time. □

Remark 4. All the relations and protocols of the section can be naturally extended into ones where the vectors consist of a public part $\mathbf{x} \in \mathbb{F}^\ell$ known to $\mathcal{V}$ and a witness part $\mathbf{w} \in \mathbb{F}^{n-\ell}$ that is committed as $[w(\mathbf{X})] = [\mathbf{w}^\top(\lambda_{\ell+1}(\mathbf{X}), \ldots, \lambda_n(\mathbf{X}))]$. The only difference is that $\mathcal{P}$ now provides evaluations $w(\boldsymbol{\alpha})$ and $\mathcal{V}$ constructs evaluations $(\mathbf{x}, \mathbf{w})^\top \boldsymbol{\lambda}(\boldsymbol{\alpha}) = x(\boldsymbol{\alpha}) + w(\boldsymbol{\alpha})$, where $x(\mathbf{X}) = \mathbf{x}^\top(\lambda_1(\mathbf{X}), \ldots, \lambda_\ell(\mathbf{X}))$. We choose not to include this generalization for simplicity of exposition.

Early-stopping version. We describe here how Π_{GBF2} can capture the PIOPs in popular state-of-the-art folding schemes [KS24, BC25]. In more detail, we define the early-stopping version Π_{esGBF2} to be the protocol that stops after the prover message in (b). Hence, Π_{esGBF2} is an interactive reduction of knowledge from $\mathcal{R}_{\mathsf{GBF}}$ to $\mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}^*_{\mathsf{GBF},\alpha}$, where each statement in $\mathcal{R}^*_{\mathsf{GBF},\alpha}$ is of the form $\zeta_{j_M,j_v} = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M} \mathbf{v}_{j_v}$ for the same $\boldsymbol{\alpha} \in \mathbb{F}^\nu$. In [KS24] this relation is called linearized committed CCS. When Π_{esGBF2} is applied to $\mathcal{R}^*_{\mathsf{GBF}}$ where all statements share the same sets $S^{(k)}_{r,i}$, then the reduced statement includes statements of the form $\zeta^{(k)}_{j_M,j_v} = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M} \mathbf{v}^{(k)}_{j_v}$, for all $k \in [K]$. In HyperNova the authors leverage homomorphic commitments to combine said statements to a single one $\zeta_{j_M,j_v} = \boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \mathbf{M}_{j_M} \mathbf{v}$, where $\zeta_{j_M,j_v} := \sum_{k\in[K]} \eta_k \zeta^{(k)}_{j_M,j_v}$ and $\mathbf{v} := \sum_{k\in[K]} \eta_k \mathbf{v}^{(k)}_{j_v}$.

5 Proof Carrying Data

In Section 5.1 we construct a family of interactive reductions of knowledge Barebones from $\mathcal{R}_{\mathsf{CCS}}$ to correctness of polynomial commitment evaluation proof and batched matrix evaluations, i.e. $\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$, using Π_{GBF} as a central component. Through the different instantiations of Π_{GBF} we recover SuperMarlin and SuperSpartan [STW23] but generalized to work for both univariate and multivariate polynomials. In Section 5.2 we construct a family of many-to-one interactive reductions of knowledge Π_{Fold} from $\left(\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}\right)^*$ to $\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ using Π_{GBF} and assuming that the PC scheme has accumulatable evaluation proofs. Finally, assuming a homomorphic PC_2 scheme, in Section 5.3 we design a succinct interactive decider whose cost is dominated by a linear combination of the matrix polynomial commitments and the verification of a single PC_2 proof.

5.1 Barebones$_\nu$

We begin by providing an auxiliary interactive reduction of knowledge Π_{Collapse} from $\mathcal{R}_{\mathsf{CCS}}$ to $\mathcal{R}_{\mathsf{GBF},\alpha}$. Π_{Collapse} produces the commitments to the matrices and collapses the n linear equations of CCS to a single one using verifier's randomness. It essentially corresponds to the preprocessing phase and first round of virtually all interactive arguments of knowledge for CCS.

Lemma 2. *There exists an interactive reduction of knowledge from $\mathcal{R}_{\mathsf{CCS}}$ to $\mathcal{R}_{\mathsf{GBF},\alpha}$, with perfect completeness, knowledge soundness error $\frac{n-1}{|\mathbb{F}|}$ if $\nu = 1$ and $\frac{\log n}{|\mathbb{F}|}$ if $\nu = \log n$, extractor time $\mathcal{O}(n)$, and the following efficiency properties: (i) round complexity:* 1*, (ii) messages sent by $\mathcal{P}$:* 1 *commitment, (iii) verifier random elements: $\nu = 1$:* 1*, $\nu = \log n$: $\log n$, and (iv) verification complexity: $\mathcal{O}(1)$.*

Proof. We construct an interactive reduction of knowledge Π_{Collapse} from $\mathcal{R}_{\mathsf{CCS}}$ to the $\mathcal{R}_{\mathsf{GBF},\alpha}$ with public inputs (see discussion in Remark 4).

	Π_{Collapse}	$\Pi_{\mathsf{GBF},\alpha}$	Π_{batchM}	Π_{provePCE}	Barebones
Rounds	$\mathsf{rc}_{\mathsf{Collapse}}$	$\mathsf{rc}_{\mathsf{GBF},\alpha}$	$\mathsf{rc}_{\mathsf{batchM}}$	$\mathsf{rc}_{\mathsf{provePCE}}$	$\sum$
Prover messages	$\mathsf{msp}_{\mathsf{Collapse}}$	$\mathsf{msp}_{\mathsf{GBF},\alpha}$	$\mathsf{msp}_{\mathsf{batchM}}$	$\mathsf{msp}_{\mathsf{provePCE}}$	$\sum$
Verifier randomness	$\mathsf{vre}_{\mathsf{Collapse}}$	$\mathsf{vre}_{\mathsf{GBF},\alpha}$	$\mathsf{vre}_{\mathsf{batchM}}$	$\mathsf{vre}_{\mathsf{provePCE}}$	$\sum$
Verification complexity	$\mathsf{vc}_{\mathsf{Collapse}}$	$\mathsf{vc}_{\mathsf{GBF},\alpha}$	$\mathsf{vc}_{\mathsf{batchM}}$	$\mathsf{vc}_{\mathsf{provePCE}}$	$\sum$
Extractor complexity	$\mathsf{et}_{\mathsf{Collapse}}$	–	–	$\mathsf{et}_{\mathsf{provePCE}}$	$\sum$
Knowledge error	$\epsilon_{\mathsf{Collapse}}$	$\epsilon_{\mathsf{GBF},\alpha}$	$\epsilon_{\mathsf{batchM}}$	$\epsilon_{\mathsf{provePCE}}$	ϵ

Table 3. Cost decomposition of Barebones. Totals are the sums of the component costs.

Π_{Collapse}: The indexer $\mathcal{I}$ on input public parameters pp and index $\mathbb{i}_{\mathsf{CCS}}$, computes commitments $[M_i(\mathbf{X},\mathbf{Y})] = \mathsf{PC}_2.\mathsf{Commit}(\mathsf{ck}_2, M_i(\mathbf{X},\mathbf{Y}), d_\nu)$, and outputs short index $\iota = (n, t_M, [M_i(\mathbf{X},\mathbf{Y})]_{i=1}^{t_M}, \mathsf{vk}, \mathsf{vk}_2)$ and index $\mathbb{i}_{\mathsf{GBF},\alpha} = (n, t_M, \mathcal{M})$. The interaction is as follows.

$\mathcal{P}$ computes $w(\mathbf{X}) = \mathbf{w}^\top(\lambda_{\ell+1}(\mathbf{X}), \ldots, \lambda_n(\mathbf{X}))$ and sends $[w(\mathbf{X})]$ to $\mathcal{V}$, who then responds with random $\boldsymbol{\alpha} \in \mathbb{F}^\nu$. They both define the $\mathcal{R}_{\mathsf{GBF},\alpha}$ with $\boldsymbol{\alpha}, \mathbf{x}, [w(\mathbf{X})]$ included in the instance that satisfies

$$\boldsymbol{\lambda}(\boldsymbol{\alpha})^\top \Big(\sum_{i\in[q]} c_i \bigcirc_{j\in S_i} \mathbf{M}_j \mathbf{z} \Big) = 0, \quad \mathbf{z} = (\mathbf{x}, \mathbf{w}).$$

The aforementioned reduction incurs knowledge soundness error $\epsilon_{\mathsf{collapse}} = \frac{n-1}{|\mathbb{F}|}$ in case $\nu = 1$ and $\frac{\log n}{|\mathbb{F}|}$, in case $\nu = \log n$. To see this, assume $(\mathbb{i}, \mathbb{x}_{\mathsf{GBF},\alpha}, \mathbb{w}_{\mathsf{GBF},\alpha}) \in \mathcal{R}_{\mathsf{GBF},\alpha}$.

The extractor $\mathcal{E}$ uses $\mathcal{E}_{\mathsf{PC}}$ to get $w(\mathbf{X})$ from $[w(\mathbf{X})]$ and computes $\mathbf{w} \in \mathbb{F}^{n-\ell}$ such that $w_i = w(\mathbf{h}_{\ell+i})$. Since $\boldsymbol{\alpha}$ was sent after the commitment to $[w(\mathbf{X})]$ we have that, except with probability $\frac{n-1}{|\mathbb{F}|}$ when $\nu = 1$ and $\frac{1}{|\mathbb{F}|}$ when $\nu = \log n$,

$$\boldsymbol{\lambda}(\mathbf{X})^\top \big(\sum_{i\in[q]} c_i \bigcirc_{j\in S_i} \mathbf{M}_j \mathbf{z} \big) = 0.$$

The above implies that $\forall i' \in [n]$, $\sum_{i\in[q]} c_i \bigcirc_{j\in S_i} \sum_{j'\in[n]} \mathbf{M}_j^{i',j'} z_{j'} = 0$, where $\mathbf{M}_j^{i',j'}$ is the (i',j') entry of matrix $\mathbf{M}_j$. Hence, we deduce that $\sum_{i\in[q]} c_i \bigcirc_{j\in S_i} \mathbf{M}_j \mathbf{z} = \mathbf{0}$. □

We can now construct the desired family of interactive reductions of knowledge Barebones as a composition of the protocols we have already constructed.

Theorem 7 (Barebones)**.** *Let Π_{Collapse}, $\Pi_{\mathsf{GBF},\alpha}$, and Π_{batchM} be the interactive reductions of knowledge defined in Lemma 2, Corollary 1, and Lemma 1, respectively. Then, there exists a family of interactive reductions of knowledge* Barebones *from $\mathcal{R}_{\mathsf{CCS}}$ to $\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$ with perfect completeness. Its efficiency parameters and round-by-round knowledge soundness are summarized in Table 3.*

	$\Pi_{\mathsf{GBF},\alpha,\beta}$	Π_{provePCE}	Π_{batchM}	$\Pi_{\mathsf{batchPCEP}}$	Π_{Fold}
Rounds	$\mathsf{rc}_{\mathsf{GBF},\alpha,\beta}$	$\mathsf{rc}_{\mathsf{provePCE}}$	$\mathsf{rc}_{\mathsf{batchM}}$	$\mathsf{rc}_{\mathsf{batchPCEP}}$	$\sum$
Prover messages	$\mathsf{msp}_{\mathsf{GBF},\alpha,\beta}$	$\mathsf{msp}_{\mathsf{provePCE}}$	$\mathsf{msp}_{\mathsf{batchM}}$	$\mathsf{msp}_{\mathsf{batchPCEP}}$	$\sum$
Verifier randomness	$\mathsf{vre}_{\mathsf{GBF},\alpha,\beta}$	$\mathsf{vre}_{\mathsf{provePCE}}$	$\mathsf{vre}_{\mathsf{batchM}}$	$\mathsf{vre}_{\mathsf{batchPCEP}}$	$\sum$
Verification complexity	$\mathsf{vc}_{\mathsf{GBF},\alpha,\beta}$	$\mathsf{vc}_{\mathsf{provePCE}}$	$\mathsf{vc}_{\mathsf{batchM}}$	$\mathsf{vc}_{\mathsf{batchPCEP}}$	$\sum$
Extractor complexity	–	$\mathsf{et}_{\mathsf{provePCE}}$	–	–	$\mathsf{et}_{\mathsf{provePCE}}$
Knowledge error	$\epsilon_{\mathsf{GBF},\alpha,\beta}$	$\epsilon_{\mathsf{provePCE}}$	$\epsilon_{\mathsf{batchM}}$	$\epsilon_{\mathsf{batchPCEP}}$	ϵ

Table 4. Cost decomposition of Π_{Fold}. Totals are the sums of the component costs.

Proof. The prover and verifier compose the component reductions along the following pipeline, where each waypoint is the working relation and each arrow is labeled by the reduction applied:

$$\begin{array}{c} \mathcal{R}_{\mathsf{CCS}} \\ \downarrow \quad \Pi_{\mathsf{Collapse}} \text{ (Lemma 2)} \\ \mathcal{R}_{\mathsf{GBF},\alpha} \\ \downarrow \quad \Pi_{\mathsf{GBF},\alpha} \text{ (Corollary 1)} \\ \mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}} \\ \downarrow \quad \mathsf{ID}_{\mathcal{R}_{\mathsf{PCE}}} \times \Pi_{\mathsf{batchM}} \text{ (Lemma 1)} \\ \mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta} \\ \downarrow \quad \Pi_{\mathsf{provePCE}} \times \mathsf{ID}_{\mathcal{R}_{\mathsf{GBF},\alpha,\beta}} \\ \mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta} \end{array}$$

The extractor $\mathcal{E}$ recovers the witness $\mathbf{w}$ via $\mathcal{E}_{\mathsf{Collapse}}$ in the first step. Composing these reductions and applying Theorem 3 yields

$$\mathsf{Barebones} := (\Pi_{\mathsf{provePCE}} \times \mathsf{ID}) \circ (\mathsf{ID} \times \Pi_{\mathsf{batchM}}) \circ \Pi_{\mathsf{GBF},\alpha} \circ \Pi_{\mathsf{Collapse}} \colon \mathcal{R}_{\mathsf{CCS}} \to \mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta},$$

with the claimed properties. □

5.2 Accumulation

We describe here the second component needed to construct PCD, namely the accumulator. Intuitively, Π_{Fold} on input multiple polynomial commitment evaluation proof and matrix polynomial evaluation statements invokes $\Pi_{\mathsf{GBF},\alpha,\beta}$ to reduce the latter to a single statement for a fresh point and $\Pi_{\mathsf{batchPCEP}}$ to batch the evaluation proofs.

Theorem 8 (Π_{Fold}). *Let $\Pi_{\mathsf{GBF},\alpha,\beta}$ and Π_{batchM} be the interactive reductions of knowledge defined in Corollary 2 and Lemma 1, respectively. Then, there exists a family of interactive reductions of knowledge Π_{Fold} from $\mathcal{R}^*_{\mathsf{PCEP}} \times \mathcal{R}^*_{\mathsf{GBF},\alpha,\beta}$ to $\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$, with perfect completeness. Its efficiency properties and round-by-round knowledge soundness are summarized in Table 4.*

Proof. The prover and verifier compose the reductions along the pipeline below, starting from $\mathcal{R}^*_{\mathsf{Acc}} = \mathcal{R}^*_{\mathsf{PCEP}} \times \mathcal{R}^*_{\mathsf{GBF},\alpha,\beta}$ and ending at $\mathcal{R}_{\mathsf{Acc}} = \mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta}$:

$$\begin{array}{ll}
\mathcal{R}^*_{\mathsf{PCEP}} \times \mathcal{R}^*_{\mathsf{GBF},\alpha,\beta} & (= \mathcal{R}^*_{\mathsf{Acc}}) \\
\downarrow & \mathsf{ID}_{\mathcal{R}_{\mathsf{PCEP}}} \times \Pi_{\mathsf{GBF},\alpha,\beta} \text{ (Corollary 2)} \\
\mathcal{R}^*_{\mathsf{PCEP}} \times \mathcal{R}^*_{\mathsf{PCE}} \times \mathcal{R}_{\mathsf{hbPCE}} & \\
\downarrow & \mathsf{ID}_{\mathcal{R}_{\mathsf{PCEP}}} \times \Pi_{\mathsf{provePCE}} \times \Pi_{\mathsf{batchM}} \text{ (Lemma 1)} \\
\mathcal{R}^*_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta} & \\
\downarrow & \Pi_{\mathsf{batchPCEP}} \times \mathsf{ID}_{\mathcal{R}_{\mathsf{GBF},\alpha,\beta}} \\
\mathcal{R}_{\mathsf{PCEP}} \times \mathcal{R}_{\mathsf{GBF},\alpha,\beta} & (= \mathcal{R}_{\mathsf{Acc}})
\end{array}$$

Composing and applying Theorem 3 yields

$$\Pi_{\mathsf{Fold}} := (\Pi_{\mathsf{batchPCEP}} \times \mathsf{ID}) \circ (\mathsf{ID} \times \Pi_{\mathsf{provePCE}} \times \Pi_{\mathsf{batchM}}) \circ (\mathsf{ID} \times \Pi_{\mathsf{GBF},\alpha,\beta}) \colon \mathcal{R}^*_{\mathsf{Acc}} \to \mathcal{R}_{\mathsf{Acc}},$$

with the claimed properties. □

Corollary 3. *There exists a family of PCD schemes in the standard model for constant-depth compliance predicates.*

Proof. Let Barebones and Π_{Fold} be the interactive reductions of knowledge from Theorems 7 and 8, respectively. Since the constructions satisfy round-by-round knowledge soundness, we can apply the Fiat-Shamir transform [FS87] and get non-interactive reductions of knowledge NI-Barebones and NI-Π_{Fold} that satisfy straightline knowledge soundness [CY24]. Next, we apply Theorem 4 to get non-interactive argument ARG, and accumulation scheme ACC for ARG in the ROM. By heuristically instantiating the ROM with an appropriate, concrete hash function we get a pair of argument and accumulation in the standard model. Now, we apply Theorem 9 to get PCD for constant-depth compliance predicates in the standard model. □

5.3 Deciding Non-Uniform PCD

In this section we present a succinct, interactive decider for the non-uniform PCD case, assuming a homomorphic commitment scheme is used for the 2ν-variate polynomials $M_i(\mathbf{X}, \mathbf{Y})$ encoding the matrices (such as the ones in [BMM+21, GPPS24, NPR24]).

To this end, let $\mathcal{F} = \{f_1, \ldots, f_\ell\}$ be the set of functions used in the non-uniform PCD and assume that for each $i \in [\ell]$, the function f_i is encoded with $t^{(i)}_M$ CCS matrices $\mathbf{M}^{(i)}_1, \ldots, \mathbf{M}^{(i)}_{t^{(i)}_M} \in \mathbb{F}^{n \times n}$. Moreover, as in SuperNova [KS22] and subsequent works [KS24, BHKZ25], we assume that the accumulator consists of a length-ℓ vector of accumulators $\mathsf{acc} = (\mathsf{acc}^{(1)}, \ldots, \mathsf{acc}^{(\ell)})$ (i.e. we keep a separate accumulator for each function). The argument proof and each of the ℓ accumulators $\mathsf{acc}^{(i)}$ consists of (1) a polynomial commitment evaluation proof as well as (2) a statement of the form $\sum_{j \in [t^{(i)}_M]} \eta^{(i)}_j M^{(i)}_j(\boldsymbol{\beta}^{(i)}, \boldsymbol{\alpha}^{(i)}) = \gamma^{(i)}$. In total, this amounts to $\ell + 1$ statements in $\mathcal{R}_{\mathsf{PCEP}}$ and $\ell + 1$ in $\mathcal{R}_{\mathsf{GBF},\alpha,\beta}$.

The decider $\mathcal{D}$ takes as input the vector of accumulators acc and the argument proof. It interacts with $\mathcal{P}$ as follows. $\mathcal{P}$ and $\mathcal{D}$ run $\Pi_{\mathsf{GBF},\alpha,\beta}$ followed by Π_{batchM} on the index containing all the matrices. This achieves reducing $\ell + 1$ batched evaluations *at different points* to a batched evaluation *at the same point*, i.e. to a statement of the form

$$\sum_{i \in [\ell]} \sum_{j \in [t^{(i)}_M]} \eta^{(i)}_j M^{(i)}_j(\boldsymbol{\beta}, \boldsymbol{\alpha}) = \gamma. \tag{17}$$

Then, $\mathcal{P}$ and $\mathcal{D}$ run Π_{provePCE} followed by $\Pi_{\mathsf{batchPCEP}}$ to reduce the multiple evaluation proofs and evaluation claims (from the invocation of $\Pi_{\mathsf{GBF},\alpha,\beta}$) to a single claim in $\mathcal{R}_{\mathsf{PCEP}}$ that $\mathcal{D}$ checks directly.

Now, in order to check equation 17 we use the homomorphic property of PC_2. In more detail, $\mathcal{P}$ computes polynomial $M(\mathbf{X},\mathbf{Y}) := \sum_{i\in[\ell]}\sum_{j\in[t_M^{(i)}]}\eta_j^{(i)}M_j^{(i)}(\mathbf{X},\mathbf{Y})$ together with an evaluation proof for the claim $M(\boldsymbol{\beta},\boldsymbol{\alpha}) = \gamma$. On the other hand, $\mathcal{D}$ computes the commitment to the polynomial $M(\mathbf{X},\mathbf{Y})$ as $[M(\mathbf{X},\mathbf{Y})] = \sum_{i\in[\ell]}\sum_{j\in[t_M^{(i)}]}\eta_j^{(i)}[M_j^{(i)}(\mathbf{X},\mathbf{Y})]$ and checks the evaluation proof provided by $\mathcal{P}$.

In total, the decider cost is dominated by a single linear combination of size equal to the total number of matrices and verifying a single evaluation proof. This is in contrast to state-of-the-art deciders such as the one in [BHKZ25] that require evaluating $\sum_{i\in[\ell]} t_M^{(i)}$ matrix polynomials.

Acknowledgment. N.P. is part of the María de Maeztu Units of Excellence Programme CEX2021-001195-M, funded by MICIU/AEI/10.13039/501100011033. Additionally, C.R. and N.P. were supported by the AEI-PID2021-128521OB-I00 grant of the Spanish Ministry of Science and Innovation, and Artemisa Chair, an initiative carried out within the framework of the funds of the Recovery, Transformation and Resilience Plan, financed by the European Union (Next Generation), the project of the Spanish Government that traces the roadmap for the modernization of the Spanish economy, the recovery of economic growth and job creation, for the solid, inclusive and resilient economic reconstruction after the COVID-19 crisis, and to respond to the challenges of the next decade.

References

ACFY24. Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. STIR: Reed-solomon proximity testing with fewer queries. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part X*, volume 14929 of *LNCS*, pages 380–413. Springer, Cham, August 2024.

ACFY25. Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: Reed-solomon proximity testing with super-fast verification. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part IV*, volume 15604 of *LNCS*, pages 214–243. Springer, Cham, May 2025.

BBB+18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society Press, May 2018.

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

BC23. Benedikt Bünz and Binyi Chen. Protostar: Generic efficient accumulation/folding for special-sound protocols. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part II*, volume 14439 of *LNCS*, pages 77–110. Springer, Singapore, December 2023.

BC25. Dan Boneh and Binyi Chen. LatticeFold+: Faster, simpler, shorter lattice-based folding for succinct proof systems. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VII*, volume 16006 of *LNCS*, pages 327–361. Springer, Cham, August 2025.

BCC+16. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 327–357. Springer, Berlin, Heidelberg, May 2016.

BCCT13. Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 111–120. ACM Press, June 2013.

BCFW26. Benedikt Bünz, Alessandro Chiesa, Giacomo Fenzi, and William Wang. Linear-time accumulation schemes. In Benny Applebaum and Huijia (Rachel) Lin, editors, *Theory of Cryptography*, pages 369–399, Cham, 2026. Springer Nature Switzerland.

BCG24. Elette Boyle, Ran Cohen, and Aarushi Goel. Breaking the $O(\sqrt{n})$-bit barrier: Byzantine agreement with polylog bits per party. *Journal of Cryptology*, 37(1):2, January 2024.

BCL+21. Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data without succinct arguments. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 681–710, Virtual Event, August 2021. Springer, Cham.

BCMS20. Benedikt Bünz, Alessandro Chiesa, Pratyush Mishra, and Nicholas Spooner. Recursive proof composition from accumulation schemes. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 1–18. Springer, Cham, November 2020.

BCR+19. Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 103–128. Springer, Cham, May 2019.

BCTV14. Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 276–294. Springer, Berlin, Heidelberg, August 2014.

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, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 649–680, Virtual Event, August 2021. Springer, Cham.

BGH19. Sean Bowe, Jack Grigg, and Daira Hopwood. Recursive proof composition without a trusted setup. IACR Cryptology ePrint Archive, Report 2019/1021, 2019.

BHKZ25. Benedikt Bünz, Hossein Hafezi, George Kadianakis, and Arantxa Zapico. KZH-Fold: Accountable voting from sublinear accumulation. In *ACM CCS 2025*. ACM, 2025. Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security.

BMM+21. Benedikt Bünz, Mary Maller, Pratyush Mishra, Nirvan Tyagi, and Psi Vesely. Proofs for inner pairing products and applications. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part III*, volume 13092 of *LNCS*, pages 65–97. Springer, Cham, December 2021.

BMNW24. Benedikt Bünz, Pratyush Mishra, Wilson Nguyen, and William Wang. Accumulation without homomorphism. Cryptology ePrint Archive, Report 2024/474, 2024.

BMNW25. Benedikt Bünz, Pratyush Mishra, Wilson Nguyen, and William Wang. Arc: Accumulation for reed-solomon codes. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VII*, volume 16006 of *LNCS*, pages 128–160. Springer, Cham, August 2025.

BMRS20. Joseph Bonneau, Izaak Meckler, Vanishree Rao, and Evan Shapiro. Coda: Decentralized cryptocurrency at scale. Cryptology ePrint Archive, Report 2020/352, 2020.

BMUS23. Marta Bellés-Muñoz, Jorge Jiménez Urroz, and Javier Silva. Revisiting cycles of pairing-friendly elliptic curves. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part II*, volume 14082 of *LNCS*, pages 3–37. Springer, Cham, August 2023.

CCDW20. Weikeng Chen, Alessandro Chiesa, Emma Dauterman, and Nicholas P. Ward. Reducing participation costs via incremental verification for ledger systems. Cryptology ePrint Archive, Report 2020/1522, 2020.

CCW19. Alessandro Chiesa, Lynn Chua, and Matthew Weidner. On cycles of pairing-friendly elliptic curves. *SIAM Journal on Applied Algebra and Geometry*, 3(2):175–192, 2019.

CFF+21. Matteo Campanelli, Antonio Faonio, Dario Fiore, Anaïs Querol, and Hadrián Rodríguez. Lunar: A toolbox for more efficient universal and updatable zkSNARKs and commit-and-prove extensions. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part III*, volume 13092 of *LNCS*, pages 3–33. Springer, Cham, December 2021.

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, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 738–768. Springer, Cham, May 2020.

COS20. Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 769–793. Springer, Cham, May 2020.

CT10. Alessandro Chiesa and Eran Tromer. Proof-carrying data and hearsay arguments from signature cards. In *Proceedings of the 1st Symposium on Innovations in Computer Science (ICS '10)*, pages 310–329, Beijing, China, 2010. Tsinghua University Press.

CTV15. Alessandro Chiesa, Eran Tromer, and Madars Virza. Cluster computing in zero knowledge. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 371–403. Springer, Berlin, Heidelberg, April 2015.

CY24. Alessandro Chiesa and Eylon Yogev. *Building Cryptographic Proofs from Hash Functions*. 2024.

DP25. Benjamin E. Diamond and Jim Posen. Succinct arguments over towers of binary fields. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part IV*, volume 15604 of *LNCS*, pages 93–122. Springer, Cham, May 2025.

EG23. Liam Eagen and Ariel Gabizon. Protogalaxy: Efficient protostar-style folding of multiple instances. IACR Cryptology ePrint Archive, Report 2023/1106, 2023.

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Berlin, Heidelberg, August 1987.

GPPS24. Chaya Ganesh, Sikhar Patranabis, Shubh Prakash, and Nitin Singh. GAPP: Generic aggregation of polynomial protocols. Cryptology ePrint Archive, Report 2024/1685, 2024.

GWC19. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019.

KP23. Abhiram Kothapalli and Bryan Parno. Algebraic reductions of knowledge. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 669–701. Springer, Cham, August 2023.

KS22. Abhiram Kothapalli and Srinath Setty. SuperNova: Proving universal machine executions without universal circuits. Cryptology ePrint Archive, Report 2022/1758, 2022.

KS23. Abhiram Kothapalli and Srinath Setty. HyperNova: Recursive arguments for customizable constraint systems. Cryptology ePrint Archive, Report 2023/573, 2023.

KS24. Abhiram Kothapalli and Srinath T. V. Setty. HyperNova: Recursive arguments for customizable constraint systems. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part X*, volume 14929 of *LNCS*, pages 345–379. Springer, Cham, August 2024.

KST22. Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part IV*, volume 13510 of *LNCS*, pages 359–388. Springer, Cham, August 2022.

KZG10. Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 177–194. Springer, Berlin, Heidelberg, December 2010.

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

NDC+24. Wilson D. Nguyen, Trisha Datta, Binyi Chen, Nirvan Tyagi, and Dan Boneh. Mangrove: A scalable framework for folding-based SNARKs. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part X*, volume 14929 of *LNCS*, pages 308–344. Springer, Cham, August 2024.

NPR24. Anca Nitulescu, Nikitas Paslis, and Carla Ràfols. FLIP-and-prove R1CS. Cryptology ePrint Archive, Report 2024/1364, 2024.

Pol22. Polygon Zero Team. Plonky2: Fast recursive arguments with FRI. https://github.com/mir-protocol/plonky2, 2022. Accessed: 2026-02-12.

RZ21. Carla Ràfols and Arantxa Zapico. An algebraic framework for universal and updatable SNARKs. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 774–804, Virtual Event, August 2021. Springer, Cham.

Set20. Srinath Setty. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 704–737. Springer, Cham, August 2020.

STW23. Srinath Setty, Justin Thaler, and Riad Wahby. Customizable constraint systems for succinct arguments. Cryptology ePrint Archive, Report 2023/552, 2023.

Val08. Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 1–18. Springer, Berlin, Heidelberg, March 2008.

ZCF24. Hadas Zeilberger, Binyi Chen, and Ben Fisch. BaseFold: Efficient field-agnostic polynomial commitment schemes from foldable codes. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part X*, volume 14929 of *LNCS*, pages 138–169. Springer, Cham, August 2024.

A Deferred Definitions and Theorems

A.1 Polynomial Commitments

We adapt the polynomial commitment definitions of [CHM+20] to the multivariate case. We slightly simplify the definition to (1) cover only deterministic commitments (2) consider only the case where all polynomial commitments are opened at the same point with the same degree bound. We assume that the zero-knowledge property of the SNARK is achieved through other means, such as extension with a dummy circuit [RZ21].

Definition 9 (Polynomial Commitment Scheme). *A polynomial commitment scheme over a field family $\mathcal{F}$ is a tuple of algorithms* $\mathsf{PC} = (\mathsf{Setup}, \mathsf{Trim}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ *with the following syntax:*

- $\mathsf{PC.Setup}\left(1^\lambda, \mathbf{D}\right) \to \mathsf{pp}$. *On input a security parameter λ (in unary), $\mu \in \mathbb{N}^+$, and a vector of maximum degree bounds $\mathbf{D} \in (\mathbb{N}^+)^\mu$, this algorithm samples public parameters* pp, *which include the description of a finite field $\mathbb{F} \in \mathcal{F}$.*
- $\mathsf{PC.Trim}^{\mathsf{pp}}\left(1^\lambda, \mathbf{d}\right) \to (\mathsf{ck}, \mathsf{vk})$. *Given oracle access to public parameters* pp, *and on input a security parameter λ (in unary), a vector of degree bounds $\mathbf{d}$, where $d_i \leq D_i$ for all $i \in [\mu]$,* PC.Trim *deterministically computes a key pair* $(\mathsf{ck}, \mathsf{vk})$ *that is specialized to $\mathbf{d}$.*
- $\mathsf{PC.Commit}(\mathsf{ck}, \boldsymbol{p}, \boldsymbol{d}) \to \boldsymbol{c}$. *On input* ck, *polynomials $\boldsymbol{p} = [p_i]_{i=1}^{n_{pc}}$ in μ variables $\mathbf{X}$ over field $\mathbb{F}$, and a vector of degree bounds $\mathbf{d}$ with $\deg_{X_i}(p_j) \leq d_i$ for all i, j,* PC.Commit *outputs commitments $\boldsymbol{c} = [c_i]_{i=1}^{n_{pc}}$ to the polynomials $\boldsymbol{p} = [p_i]_{i=1}^{n_{pc}}$.*
- $\mathsf{PC.Open}(\mathsf{ck}, \boldsymbol{p}, \mathbf{d}, Q, \xi) \to \pi$. *On input* ck, *polynomials $\boldsymbol{p} = [p_i]_{i=1}^{n_{pc}} \in \mathbb{F}[\mathbf{X}]$ with degree bound $\mathbf{d}$, a query set Q consisting of tuples $(i, \mathbf{z}) \in [n_{pc}] \times \mathbb{F}^\mu$, and opening challenge ξ,* PC.Open *outputs an evaluation proof π.*
- $\mathsf{PC.Verify}(\mathsf{vk}, \boldsymbol{c}, \boldsymbol{d}, Q, \boldsymbol{v}, \pi, \xi) \in \{0, 1\}$. *On input* vk, *commitments $\boldsymbol{c} = [c_i]_{i=1}^{n_{pc}}$, degree bounds $\boldsymbol{d} = [d_i]_{i=1}^{n_{pc}}$, query set Q consisting of tuples $(i, \mathbf{z}) \in [n_{pc}] \times \mathbb{F}^\mu$, alleged evaluations $\boldsymbol{v} = \left(v_{(i,\mathbf{z})}\right)_{(i,z)\in Q}$, evaluation proof π, and opening challenge ξ,* PC.Verify *outputs 1 if π attests that, for every $(i, \mathbf{z}) \in Q$, the polynomial p_i committed in c_i has degree at most $\mathbf{d}$ and evaluates to $v_{(i,\mathbf{z})}$ at $\mathbf{z}$.*

A polynomial commitment scheme PC must satisfy the completeness property defined below. Extractability is also a requirement if one wants to use polynomial commitments to "compile" information theoretic objects into SNARKs. In that case, commitments to polynomials might be defined in several rounds and the definition of extractability captures this.

Extractability. For every maximum degree bound $\mathbf{D} \in \mathbb{N}$ and efficient adversary $\mathcal{A}$, there exists an efficient extractor $\mathcal{E}$ such that for every round bound $\mathsf{r} \in \mathbb{N}$, efficient public-coin challenger $\mathcal{C}$ (each of its messages is a uniformly random string of prescribed length, or an empty string), efficient query sampler $\mathcal{Q}$, and efficient adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$:

where we assume that $\mathcal{A}, \mathcal{Q}, \mathcal{B}$ share the same random string to win the game.

A.2 Arguments and Accumulators

Definition 10. *A (preprocessing)* ***non-interactive argument*** *in the random oracle model is a tuple of polynomial time algorithms* $\mathsf{ARG} = (\mathcal{G}, \mathcal{I}, \mathcal{P}, \mathcal{V})$ *that satisfy the following properties.*

Completeness. ARG *is complete if the following holds. For every adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{c|l} (\mathbf{i}, \mathbf{x}, \mathbf{w}) \in \mathcal{R}^\rho(\mathsf{pp}) & \rho \leftarrow \mathcal{U}(\lambda) \\ \Downarrow & \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda) \\ \mathcal{V}^\rho(\mathsf{vk}, \mathbf{x}, \pi) = 1 & (\mathbf{i}, \mathbf{x}, \mathbf{w}) \leftarrow \mathcal{A}^\rho(\mathsf{pp}) \\ & (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{I}^\rho(\mathsf{pp}, \mathbf{i}) \\ & \pi \leftarrow \mathcal{P}^\rho(\mathsf{pk}, \mathbf{x}, \mathbf{w}) \end{array}\right] = 1,$$

$$\Pr\left[\begin{array}{c|r} \mathsf{PC.Verify}\,(\mathsf{vk}, \boldsymbol{c}, \boldsymbol{d}, Q, \boldsymbol{v}, \pi, \xi) = 1 & \mathsf{pp} \leftarrow \mathsf{PC.Setup}(1^\lambda, \mathbf{D}) \\ & \hline \\ & \text{For } i = 1 \ldots \mathsf{r}: \\ & \rho_i \leftarrow \mathcal{C}(\mathsf{pp}, i) \\ & (\boldsymbol{c}_i, \mathbf{d}) \leftarrow \mathcal{A}(\mathsf{pp}, [\rho_j]_{j=1}^{i}) \\ & \boldsymbol{p}_i \leftarrow \mathcal{E}(\mathsf{pp}, [\rho_j]_{j=1}^{i}) \\ & \hline \\ \Downarrow & Q \leftarrow \mathcal{Q}(\mathsf{pp}, [\rho_j]_{j=1}^{\mathsf{r}}) \\ & (\boldsymbol{v}, \mathsf{st}) \leftarrow \mathcal{B}_1(\mathsf{pp}, [\rho_j]_{j=1}^{\mathsf{r}}, Q) \\ \deg_{X_i}(\boldsymbol{p}) \leq d_i \leq D_i \text{ and } \boldsymbol{v} = \boldsymbol{p}(Q) & \text{Sample opening challenge } \xi \\ & \pi \leftarrow \mathcal{B}_2(\mathsf{st}, \xi) \\ & \text{Set } [c_i]_{i=1}^{n_{pc}} := [\boldsymbol{c}_i]_{i=1}^{\mathsf{r}}, [p_i]_{i=1}^{n_{pc}} := [\boldsymbol{p}_i]_{i=1}^{\mathsf{r}} \\ & (\mathsf{ck}, \mathsf{vk}) \leftarrow \mathsf{PC.Trim}^{\mathsf{pp}}\left(1^\lambda, \boldsymbol{d}\right) \\ & T := \{i \in [n_{pc}] \mid (i, \mathbf{z}) \in Q\} \\ & \text{Set } \boldsymbol{c} := [c_i]_{i \in T}, \boldsymbol{p} := [p_i]_{i \in T} \end{array}\right] = 1 - \mathsf{negl}(\lambda),$$

Knowledge soundness. ARG *is knowledge sound (with respect to auxiliary input distribution $\mathcal{D}$) if the following holds. There exists a deterministic polynomial-time extractor $\mathcal{E}$ such that for every (non-uniform) polynomial-time adversary $\tilde{\mathcal{P}}$,*

$$\Pr\left[\begin{array}{c|l} \mathcal{V}^\rho(\mathsf{vk}, \mathbf{x}, \pi) = 1 & \rho \leftarrow \mathcal{U}(\lambda) \\ \wedge & \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda) \\ (\mathbf{i}, \mathbf{x}, \mathbf{w}) \notin \mathcal{R}^\rho(\mathsf{pp}) & \mathsf{ai} \leftarrow \mathcal{D}(1^\lambda) \\ & (\mathbf{i}, \mathbf{x}, \pi; \mathsf{tr}) \leftarrow \tilde{\mathcal{P}}^\rho(\mathsf{pp}, \mathsf{ai}) \\ & (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{I}^\rho(\mathsf{pp}, \mathbf{i}) \\ & \mathbf{w} \leftarrow \mathcal{E}(\mathsf{pp}, \mathbf{i}, \mathbf{x}, \pi, \mathsf{ai}, \mathsf{tr}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 11. *Let* $\mathsf{ARG} = (\mathcal{G}, \mathcal{I}, \mathcal{P}, \mathcal{V})$ *be a non-interactive argument in the random oracle model for index relation $\mathcal{R}^\rho$ such that the proofs have a canonical partition into pairs $\pi := (\pi.\mathbf{x}, \pi.\mathbf{w})$. An (split)* ***accumulation scheme*** ACC *for* ARG *in the random oracle model is a tuple of polynomial-time algorithms* $\mathsf{ACC} = (\mathcal{G}, \mathrm{I}, \mathrm{P}, \mathrm{V}, \mathrm{D})$*, which shares the same generator algorithm as* ARG*. An accumulation scheme must satisfy the following properties.*
Completeness. ACC *is complete if the following holds. For every adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{c|c} \forall i \in [n], \mathcal{V}^\rho(\mathsf{vk}, \mathbf{x}_i, \pi_i) = 1 \wedge & \rho \leftarrow \mathcal{U}(\lambda) \\ \forall j \in [m], \mathrm{D}(\mathsf{dk}, \mathsf{acc}_j) = 1 & \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda) \\ \Downarrow & (\mathbf{i}, [\mathbf{x}_i, (\pi_i.\mathbf{x}, \pi_i.\mathbf{w})]_{i \in [n]}, [\mathsf{acc}_j]_{j \in [m]}) \leftarrow \mathcal{A}^\rho(\mathsf{pp}) \\ \mathrm{V}\left(\begin{array}{c} \mathsf{avk}, [\mathbf{x}_i, \pi_i.\mathbf{x}]_{i \in [n]}, \\ [\mathsf{acc}_j.\mathbf{x}]_{j \in [m]}, \mathsf{acc}.\mathbf{x}, \mathsf{pf} \end{array}\right) = 1 & (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{I}^\rho(\mathsf{pp}, \mathbf{i}) \\ & (\mathsf{apk}, \mathsf{avk}, \mathsf{dk}) \leftarrow \mathrm{I}^\rho(\mathsf{pp}, \mathbf{i}) \\ \wedge \mathrm{D}(\mathsf{dk}, \mathsf{acc}) = 1 & (\mathsf{acc}, \mathsf{pf}) \leftarrow \mathrm{P}^\rho(\mathsf{apk}, [\mathbf{x}_i, \pi_i]_{i \in [n]}, [\mathsf{acc}_j]_{j \in [m]}) \end{array}\right] = 1,$$

where each accumulator has a canonical partition into pairs $\mathsf{acc} := (\mathsf{acc}.\mathbf{x}, \mathsf{acc}.\mathbf{w})$.
Knowledge soundness. ACC *is knowledge sound (with respect to auxiliary input distribution $\mathcal{D}$) if there exists a deterministic polynomial-time extractor* E *such that for every (non-uniform) polynomial-time adversary* $\tilde{\mathrm{P}}$*, the following holds*

$$\Pr\left[\begin{array}{c} \mathrm{V}\left(\begin{array}{c}\mathsf{avk}, [\mathbf{x}_i, \pi_i.\mathbf{x}]_{i\in[n]},\\ [\mathsf{acc}_j.\mathbf{x}]_{j\in[m]}, \mathsf{acc}.\mathbf{x}, \mathsf{pf}\end{array}\right) = 1 \\ \wedge \mathrm{D}(\mathsf{dk}, \mathsf{acc}) = 1 \\ \wedge \\ \exists i \in [n], \mathcal{V}^{\rho}(\mathsf{vk}, \mathbf{x}_i, \pi_i) \neq 1 \\ \vee \exists j \in [m], \mathrm{D}(\mathsf{dk}, \mathsf{acc}_j) \neq 1 \end{array} \middle| \begin{array}{c} \rho \leftarrow \mathcal{U}(\lambda) \\ \mathsf{pp} \leftarrow \mathcal{G}(1^{\lambda}) \\ \mathsf{ai} \leftarrow \mathcal{D}(1^{\lambda}) \\ \left(\begin{array}{c}\mathbf{i}, [\mathbf{x}_i, \pi_i.\mathbf{x}]_{i\in[n]},\\ [\mathsf{acc}_j.\mathbf{x}]_{j\in[m]}, \mathsf{acc}, \mathsf{pf}\end{array}; \mathsf{tr}\right) \leftarrow \tilde{\mathrm{P}}^{\rho}(\mathsf{pp}, \mathsf{ai}) \\ (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{I}^{\rho}(\mathsf{pp}, \mathbf{i}) \\ (\mathsf{apk}, \mathsf{avk}, \mathsf{dk}) \leftarrow \mathrm{I}^{\rho}(\mathsf{pp}, \mathbf{i}) \\ \left([\pi_i.\mathbf{w}]_{i\in[n]}, [\mathsf{acc}_j.\mathbf{w}]_{j\in[m]}\right) \\ \leftarrow \mathrm{E}\left(\mathsf{pp}, \begin{array}{c}\mathbf{i}, [\mathbf{x}_i, \pi_i.\mathbf{x}]_{i\in[n]},\\ [\mathsf{acc}_j.\mathbf{x}]_{j\in[m]}, \mathsf{acc}, \mathsf{pf}\end{array}, \mathsf{ai}; \mathsf{tr}\right) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

A.3 Proof-Carrying Data

A tuple of algorithms $\mathsf{PCD} = (\mathbb{G}, \mathbb{I}, \mathbb{P}, \mathbb{V})$ is a (preprocessing) proof-carrying data scheme (PCD scheme) for a class of compliance predicates F if the properties below hold.

Definition 12. *A transcript T is a directed acyclic graph where each vertex $u \in V(\mathsf{T})$ is labeled by local data $z^{(u)}_{\mathsf{loc}}$ and each edge $e \in E(\mathsf{T})$ is labeled by a message $z^{(e)} \neq \perp$. The output of a transcript T, denoted $\mathsf{o}(\mathsf{T})$, is $z^{(e)}$ where $e = (u, v)$ is the lexicographically-first edge such that v is a sink.*

Definition 13. *A vertex $u \in V(\mathsf{T})$ is φ-compliant for $\varphi \in \mathsf{F}$ if for all outgoing edges $e = (u, v) \in E(\mathsf{T})$:*

(i) (base case) if u has no incoming edges, $\varphi\left(z^{(e)}, z^{(u)}_{\mathsf{loc}}, \perp, \ldots, \perp\right)$ accepts;

(ii) (recursive case) if u has incoming edges $e_1, \ldots, e_m$, $\varphi\left(z^{(e)}, z^{(u)}_{\mathsf{loc}}, z^{(e_1)}, \ldots, z^{(e_m)}\right)$ accepts.

We say that T is φ-compliant if all of its vertices are φ-compliant.

Completeness. PCD has perfect completeness if for every adversary $\mathcal{A}$ the following holds:

$$\Pr\left[\begin{array}{c}\left(\begin{array}{c}\varphi \in \mathsf{F} \\ \wedge \varphi\left(z, z_{\mathrm{loc}}, z_1, \ldots, z_m\right) = 1 \\ \wedge \left(\forall i, z_i = \perp \vee \forall i, \mathbb{V}\left(\mathsf{ivk}, z_i, \pi_i\right) = 1\right) \\ \Downarrow \\ \mathbb{V}(\mathsf{ivk}, z, \pi) = 1\end{array}\right)\end{array}\middle|\begin{array}{r}\mathbb{pp} \leftarrow \mathbb{G}\left(1^{\lambda}\right) \\ \left(\varphi, z, z_{\mathrm{loc}}, [z_i, \pi_i]_{i=1}^{m}\right) \leftarrow \mathcal{A}(\mathsf{pp}) \\ (\mathsf{ipk}, \mathsf{ivk}) \leftarrow \mathbb{I}(\mathsf{pp}, \varphi) \\ \pi \leftarrow \mathbb{P}\left(\mathsf{ipk}, z, z_{\mathrm{loc}}, [z_i, \pi_i]_{i=1}^{m}\right)\end{array}\right] = 1.$$

Knowledge soundness. PCD has knowledge soundness (with respect to auxiliary input distribution $\mathcal{D}$) if for every expected polynomial-time adversary $\tilde{\mathbb{P}}$ there exists an expected polynomial-time extractor $\mathbb{E}_{\tilde{\mathbb{P}}}$ such that for every set Z,

$$\begin{aligned}&\Pr\left[\begin{array}{c}\varphi \in \mathsf{F} \\ \wedge(\mathbb{pp}, \mathsf{ai}, \varphi, \mathsf{o}(\mathsf{T}), \mathsf{ao}) \in Z \\ \wedge \mathsf{T} \text{ is } \varphi\text{-compliant}\end{array}\middle|\begin{array}{c}\mathbb{pp} \leftarrow \mathbb{G}\left(1^{\lambda}\right) \\ \mathsf{ai} \leftarrow \mathcal{D}(\mathbb{pp}) \\ (\varphi, \mathsf{T}, \mathsf{ao}) \leftarrow \mathbb{E}_{\tilde{\mathbb{P}}}(\mathbb{pp}, \mathsf{ai})\end{array}\right] \\ &\geq \Pr\left[\begin{array}{c}\varphi \in \mathsf{F} \\ \wedge(\mathbb{pp}, \mathsf{ai}, \varphi, \mathsf{o}, \mathsf{ao}) \in Z \\ \wedge \mathbb{V}(\mathsf{ivk}, \mathsf{o}, \pi) = 1\end{array}\middle|\begin{array}{r}\mathbb{pp} \leftarrow \mathbb{G}\left(1^{\lambda}\right) \\ \mathsf{ai} \leftarrow \mathcal{D}(\mathbb{pp}) \\ (\varphi, \mathsf{o}, \pi, \mathsf{ao}) \leftarrow \mathbb{P}(\mathbb{pp}, \mathsf{ai}) \\ (\mathsf{ipk}, \mathsf{ivk}) \leftarrow \mathbb{I}(\mathbb{pp}, \varphi)\end{array}\right] - \mathrm{negl}(\lambda).\end{aligned}$$

Zero knowledge. PCD has (statistical) zero knowledge if there exists a probabilistic polynomial-time simulator $\mathbb{S}$ such that for every honest adversary $\mathcal{A}$ the distributions below are statistically close:

$$\left\{ (\mathbb{pp}, \varphi, z, \pi) \;\begin{array}{r} \mathbb{pp} \leftarrow \mathbb{G}\left(1^\lambda\right) \\ \left(\varphi, z, z_{\mathsf{loc}}, [z_i, \pi_i]_{i=1}^m\right) \leftarrow \mathcal{A}(\mathbb{pp}) \\ (\mathsf{ipk}, \mathsf{ivk}) \leftarrow \mathbb{I}(\mathbb{pp}, \varphi) \\ \pi \leftarrow \mathbb{P}\left(\mathsf{ipk}, z, z_{\mathsf{loc}}, [z_i, \pi_i]_{i=1}^m\right) \end{array} \right\} \text{ and } \left\{ (\mathbb{pp}, \varphi, z, \pi) \;\begin{array}{r} (\mathbb{pp}, \tau) \leftarrow \mathbb{S}(1^\lambda) \\ \left(\varphi, z, z_{\mathsf{loc}}, [z_i, \pi_i]_{i=1}^m\right) \leftarrow \mathcal{A}(\mathbb{pp}) \\ \pi \leftarrow \mathbb{S}(\tau, \varphi, z) \end{array} \right\}$$

An adversary is honest if its output satisfies the implicant of the completeness condition with probability 1 , namely: $\varphi \in \mathsf{F}, \varphi\left(z, z_{\mathsf{loc}}, z_1, \ldots, z_m\right) = 1$, and either $\forall i, z_i = \perp$ or $\forall i, \mathbb{V}\left(\mathsf{ivk}, z_i, \pi_i\right) = 1$.

Efficiency. The generator $\mathbb{G}$, prover $\mathbb{P}$, indexer $\mathbb{I}$, and verifier $\mathbb{V}$ run in polynomial time. A proof π has size $\mathrm{poly}(\lambda, |\varphi|)$; in particular, it is not permitted to grow with each application of $\mathbb{P}$.

For completeness, we restate the key results of [BCL+21], which builds on [BCMS20] to construct proof carrying data from any NARK (i.e. an argument that is not necessarily succcint) and a split accumulation scheme.

Theorem 9. *There exists a polynomial-time transformation T such that if $\mathsf{ARG} = (\mathcal{G}, \mathcal{I}, \mathcal{P}, \mathcal{V})$ is a NARK for circuit satisfiability and AS is a split accumulation scheme for ARG then $\mathsf{PCD} = (\mathbb{G}, \mathbb{I}, \mathbb{P}, \mathbb{V}) := T(\mathsf{ARG}, \mathsf{AS})$ is a PCD scheme for constant-depth compliance predicates, provided*

$\exists \epsilon \in (0,1)$ and a polynomial α s.t.

$$\mathrm{v}(\lambda, m, N, |\mathsf{avk}(\lambda, m, N)| + |\mathsf{acc.x}(\lambda, m, N)| + \ell) = O\left(N^{1-\epsilon} \cdot \alpha(\lambda, m, \ell)\right),$$

where

- *ℓ is some data that is given as input to the PCD verifier,*
- *$\mathrm{v}(\lambda, m, N, k)$ is the size of the circuit $\mathrm{V}^{(\lambda, m, N, k)}$, and $\mathrm{V}^{(\lambda, m, N, k)}$ refers to the circuit representation of the accumulator verifier V with input size appropriate for security parameter λ, when accumulating m number of instance-proof pairs and accumulators, for an index of size N, and predicate input size k.*
- *$|\mathsf{avk}(\lambda, m, N)|$ is the size of the accumulator verification key avk,*
- *$|\mathsf{acc.x}(\lambda, m, N)|$ is the size of an accumulator instance.*

Moreover:

- *If ARG and AS are secure against quantum adversaries, then PCD is secure against quantum adversaries.*
- *If ARG and AS are (post-quantum) zero knowledge, then PCD is (post-quantum) zero knowledge.*
- *If the size of the predicate $\varphi : \mathbb{F}^{(m+2)\ell} \to \mathbb{F}$ is $f = \omega\left(\alpha(\lambda, m, \ell)^{1/\epsilon}\right)$ then:*
  - *the cost of running $\mathbb{I}$ is equal to the cost of running both $\mathcal{I}$ and I on an index of size $f + o(f)$;*
  - *the cost of running $\mathbb{P}$ is equal to the cost of accumulating m instance-proof pairs using P, and running $\mathcal{P}$, on an index of size $f + o(f)$ and instance of size $o(f)$;*
  - *the cost of running $\mathbb{V}$ is equal to the cost of running both $\mathcal{V}$ and D on an index of size $f + o(f)$ and an instance of size $o(f)$.*