Polynomial Spaces: A New Framework for Composite-to-Prime-Order Transformations*

Gottfried Herold[1], Julia Hesse[2], Dennis Hofheinz[2], Carla Ràfols[1], and Andy Rupp[2]

[1]Ruhr-University Bochum, Germany
{gottfried.herold,carla.rafols}@rub.de
[2]Karlsruhe Institute of Technology, Germany
{julia.hesse,dennis.hofheinz,andy.rupp}@kit.edu

Abstract

At Eurocrypt 2010, Freeman presented a framework to convert cryptosystems based on composite-order groups into ones that use prime-order groups. Such a transformation is interesting not only from a conceptual point of view, but also since for relevant parameters, operations in prime-order groups are faster than composite-order operations by an order of magnitude. Since Freeman's work, several other works have shown improvements, but also lower bounds on the efficiency of such conversions.

In this work, we present a new framework for composite-to-prime-order conversions. Our framework is in the spirit of Freeman's work; however, we develop a different, "polynomial" view of his approach, and revisit several of his design decisions. This eventually leads to significant efficiency improvements, and enables us to circumvent previous lower bounds. Specifically, we show how to verify Groth-Sahai proofs in a prime-order environment (with a symmetric pairing) almost twice as efficiently as the state of the art.

We also show that our new conversions are optimal in a very broad sense. Besides, our conversions also apply in settings with a multilinear map, and can be instantiated from a variety of computational assumptions (including, e.g., the k-linear assumption).

Keywords: bilinear maps, composite-order groups, Groth-Sahai proofs.

1 Introduction

Motivation. Cyclic groups are a very popular platform for cryptographic constructions. Starting with Diffie and Hellman's seminal work [7], there are countless examples of cryptographic schemes that work in any finite, cyclic group G, and whose security can be reduced to a well-defined computational problem in G. In many cases, the order of the group G should be prime (or is even irrelevant). However, some constructions (e.g., [12, 3, 4, 17, 25, 20]) explicitly require a group G of *composite* order.

In particular in combination with a pairing (i.e., a bilinear map) e, groups of composite order exhibit several interesting properties. (For instance, $e(g_1, g_2) = 1$ for elements g_1, g_2 of coprime order. Or, somewhat more generally, the pairing operation operates on the different prime-order components of G independently.) This enables interesting technical applications (e.g., [25, 20]), but also comes at a price. Namely, to accommodate suitably hard computational problems, composite-order groups have to be chosen substantially larger than prime-order groups. Specifically, it should be hard to factor the group order. This leads to significantly slower operations in composite-order groups: [10] suggests that for realistic parameters, Tate pairings in composite-order groups are by a factor of about 50 less efficient than in prime-order groups.

Freeman's composite-order-to-prime-order transformation. It is thus interesting to try to find substitutes for the technical features offered by composite-order groups in prime-order settings. In fact,

*An extended abstract of this work will appear in the proceedings of CRYPTO 2014. This is the full version.

Freeman [10] has offered a framework and tools to semi-generically convert cryptographic constructions from a composite-order to a prime-order setting. Similar transformations have also been implicit in previous works [13, 25]. The premise of Freeman's approach is that composite-order group elements "behave as" vectors over a prime field. In this interpretation, composite-order subgroups correspond to linear subspaces.

Moreover, we can think of the vector components as exponents of prime-order group elements; we can then associate, e.g., a composite-order subgroup indistinguishability problem with the problem of distinguishing vectors (chosen either from a subspace or the whole space) "in the exponent." More specifically, Freeman showed that the composite-order subgroup indistinguishability assumption can be implemented in a prime-order group with the Decisional Diffie-Hellman (or with the k-linear) assumption. A pairing operation over the composite-order group then translates into a suitable "multiplication of vectors," which can mean different things, depending on the desired properties. For instance, Freeman considers both an inner product and a Kronecker product as "vector multiplication" operations (of course with different effects).

Limitations of Freeman's approach. Freeman's work has spawned a number of follow-up results that investigate more general or more efficient conversions of this type [20, 22, 21, 18, 19]. We note that all of these works follow Freeman's interpretation of vectors, and even his possible interpretations of a vector multiplication. Unfortunately, during these investigations, certain lower bounds for the efficiency of these transformations became apparent. For example, Seo [21] proves lower bounds both for the computational cost and the dimension of the resulting vector space of *arbitrary* transformations in Freeman's framework. More specifically, Seo reports a concrete bound on the number of required prime-order pairing operations necessary to simulate a composite-order pairing.

However, of course, these lower bounds crucially use the vector-space interpretation of Freeman's framework. Specifically, it is conceivable that a (perhaps completely different) more efficient composite-order-to-prime-order transformation exists outside of Freeman's framework. Such a more efficient transformation could also provide a way to implement, e.g., the widely used Groth-Sahai proof system [13] more efficiently.

Our contribution: a different view on composite-order-to-prime-order conversions. In this work, we take a step back and question several assumptions that are implicitly made in Freeman's framework. We exhibit a different composite-order-to-prime-order conversion outside of his model, and show that it circumvents previous lower bounds. In particular, our construction leads to more efficient Groth-Sahai proofs in the symmetric setting (i.e., with a symmetric pairing). Moreover, our construction can be implemented from *any* matrix assumption [9] (including the k-linear assumption) and scales better to multilinear settings than previous approaches. In the following, we give more details on our construction and its properties.

A technical perspective: a polynomial interpretation of linear subspaces. To explain our approach, recall that Freeman identifies a composite-order group with a vector space over a prime field. Moreover, in his work, subgroups of the composite-order group always correspond to *uniformly chosen* subspaces of a certain dimension. Of course, such "unstructured" subspaces only allow for rather generic interpretations of composite-order pairings (as generic "vector multiplications" as above).

Instead, we interpret the composite-order group as a very structured vector space. More concretely, we interpret a composite-order group element as (the coefficient vector of) a polynomial $f(X)$ over a prime field. In this view, a composite-order subgroup corresponds to the set of all polynomials with a common zero s (for a fixed and hidden s). Composite-order group operation and pairing correspond to polynomial addition and multiplication. Moreover, the hidden common zero s can be used as a trapdoor to decide subgroup membership, and thus to implement a "projection" in the sense of Freeman.

Specifically, our "vector multiplication" is very structured and natural, and there are several ways to implement it efficiently. For instance, we can apply a convolution on the coefficient vectors, or, more efficiently, we can represent f as a vector of evaluations $f(i)$ at sufficiently many fixed values i, and multiply these evaluation vectors component-wise. In particular, we circumvent the mentioned lower bound of Seo [21] by our different interpretation of composite-order group elements as vectors.

Another interesting property of our construction is that it scales better to the multilinear setting than previous approaches. For instance, while it seems possible to generalize at least Freeman's approach to a "projecting pairing" to a setting with a k-linear map (instead of a pairing), the corresponding generic vector

multiplication would lead to exponentially (in k) large vectors in the target group. In our case, a k-linear map corresponds to the multiplication of k polynomials, and only requires a quadratic number of group elements in the target group.[1]

In the description above, f is always a univariate polynomial. With this interpretation, we can show that the SCasc assumption from Escala et al. [9] implies subgroup indistinguishability. However, we also provide a "multivariate" variant of our approach (with polynomials f in several variables) that can be implemented with *any* matrix assumption (such as the k-linear and even weaker assumptions). Furthermore, in the terminology of Freeman, we provide both a "projecting," and a "projecting and canceling" pairing construction (although the security of the "projecting and canceling" construction requires additional complexity assumptions).

Applications. The performance improvements of our approach are perhaps best demonstrated by the case of Groth-Sahai proofs. Compared to the most efficient previous implementations of Groth-Sahai proofs in prime-order groups with symmetric pairing [22, 9], we almost halve the number of required prime-order pairing operations (cf. Tab. 1). As a bonus, we also improve on the size of prime-order group elements in the target group, while retaining the small common reference string from [9].

Additionally, we show how to implement a variant of the Boneh-Goh-Nissim encryption scheme [3] in prime-order groups with a k-linear map. As already sketched, this is possible with Freeman's approach only for logarithmically small k.

Structural results. Of course, a natural question is whether *our* results are optimal, and if so, in what sense exactly. We can settle this question, in the following sense: we show that the construction sketched above is optimal in our generalized framework. We also prove a similar result for our construction from general matrix assumptions.

Open problems. In this work, we focus on settings with a *symmetric* pairing (resp. multilinear map). It is an interesting open problem to extend our approach to asymmetric settings. Furthermore, the conversion that leads to a canceling *and* projecting map (in the terminology of Freeman) requires a nonstandard complexity assumption (that however holds generically, as we prove). It would be interesting to find constructions from more standard assumptions.

Outline. After recalling some preliminaries in Sec. 2, we describe our framework in Sec. 3. Our conversions follow in Sec. 4. We discuss the optimality of our conversions in Sec. 5, and compare them to previous conversions in Sec. 6. Finally, discuss in Sec. 7 how our results imply more efficient Groth-Sahai proofs. In the appendix, we provide more detailed explanations and proofs where none could be given in the main part due to lack of space. Specifically, Appendix C discusses in detail the efficiency of our constructions. Furthermore, Sec. F.2 shows how to derive a prime-order instantiation of the Boneh-Goh-Nissim cryptosystem using our conversion, and Appendix H discusses compatibility with the recent approximate multilinear maps.

2 Preliminaries

Notation. Throughout the paper we will use additive notation for all groups G. Nevertheless, we still talk about an exponentiation with exponent a considering a scalar multiplication $a\mathcal{P}$ for $\mathcal{P} \in G$ and $a \in \mathbb{Z}_{|G|}$. Let G be a cyclic group of order p generated by $\mathcal{P}$. Then by $[a] := a\mathcal{P}$ we denote the *implicit representation* of $a \in \mathbb{Z}_p$ in G. To distinguish between implicit representations in the domain G and the target group G_T of a multilinear map we use $[\cdot]$ and $[\cdot]_T$, respectively. More generally, we also define such representations for vectors $\vec{f} \in \mathbb{Z}_p^n$ by $[\vec{f}] := ([f_i])_i \in G^n$, for matrices $\mathbf{A} = (a_{i,j})_{i,j} \in \mathbb{Z}_p^{n\times m}$ by $[\mathbf{A}] := ([a_{i,j}])_{i,j} \in G^{n\times m}$, and for sets $H \subset \mathbb{Z}_p^n$ by $[H] := \{[a] \mid a \in H\} \subset G^n$. Furthermore, we will often identify $\vec{f} \in \mathbb{Z}_p^n$ with the coefficients of a polynomial f in some space V with respect to a (fixed) basis $\mathfrak{q}_0, \ldots, \mathfrak{q}_{n-1}$ of V, i.e., $f = \sum_{i=0}^{n-1} f_i\mathfrak{q}_i$ (e.g., $V = \{f \mid f \in \mathbb{Z}_p[X], \deg(f) < n\}$ and $\mathfrak{q}_i = X^i$). In this case we may also write $[f] := [\vec{f}]$.

[1] We multiply k polynomials, and each polynomial should be of degree at least k, in order to allow for suitable subgroup indistinguishability problems that are plausible even in face of a k-linear map.

Symmetric prime-order k-linear group generators. We use the following formal definition of a k-linear prime-order group generator as the foundation for our constructions. In the scope of these constructions, we will refer to the output of such a generator as a *basic* (or, *prime-order*) k-linear map.

Definition 1 (symmetric prime-order k-linear group generator)**.** *A* symmetric prime-order k-linear group generator *is a PPT algorithm $\mathcal{G}_k$ that on input of a security parameter 1^λ outputs a tuple of the form*

$$\mathcal{MG}_k := (k, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_k(1^\lambda)$$

where G, G_T are descriptions of cyclic groups of prime order p, $\log p = \Theta(\lambda)$, $\mathcal{P}$ is a generator of G, and $e\colon G \times \ldots \times G \to G_T$ is a map which satisfies the following properties:

- ***k-linearity:*** *For all $Q_1, \ldots, Q_k \in G$, $\alpha \in \mathbb{Z}_p$, and $i \in \{1, \ldots, k\}$ we have $e(Q_1, \ldots, \alpha Q_i, \ldots, Q_k) = \alpha e(Q_1, \ldots, Q_k)$.*
- ***Non-Degeneracy:*** *$\mathcal{P}_T = e(\mathcal{P}, \ldots, \mathcal{P})$ generates G_T.*

In our paper, one should think of $\mathcal{G}_k$ as either a generator of a bilinear group setting (for $k = 2$) defined over some group of points of an elliptic curve and the multiplicative group of a finite field or, for $k > 2$, as generator of an abstract ideal multilinear map, approximated by the recent candidate constructions [11, 6].

Matrix assumptions. Our constructions are based on matrix assumptions as introduced in [9].

Definition 2 (Matrix Distributions and Assumptions [9])**.** *Let $n, \ell \in \mathbb{N}$, $n > \ell$. We call $\mathcal{D}_{n,\ell}$ a* matrix distribution *if it outputs (in probabilistic polynomial time, with overwhelming probability) matrices $\mathbf{A} \in \mathbb{Z}_p^{n \times \ell}$ of full rank ℓ. $\mathcal{D}_{n,\ell}$ is called* polynomially induced *if it is defined by picking $\vec{s} \in \mathbb{Z}_p^d$ uniformly at random and setting $a_{i,j} := \mathfrak{p}_{i,j}(\vec{s})$ for some polynomials $\mathfrak{p}_{i,j} \in \mathbb{Z}_p[\vec{X}]$ whose degrees do not depend on the security parameter. We define $\mathcal{D}_\ell := \mathcal{D}_{\ell+1,\ell}$. Furthermore, we say that the $\mathcal{D}_{n,\ell}$-*Matrix Diffie-Hellman assumption or just $\mathcal{D}_{n,\ell}$ assumption for short *holds relative to the k-linear group generator $\mathcal{G}_k$ if for all PPT adversaries* D *we have*

$$\mathbf{Adv}_{\mathcal{D}_{n,\ell},\mathcal{G}_k}(\mathsf{D}) = \mathbf{Pr}[\mathsf{D}(\mathcal{MG}_k, [\mathbf{A}], [\mathbf{A}\vec{w}]) = 1] - \mathbf{Pr}[\mathsf{D}(\mathcal{MG}_k, [\mathbf{A}], [\vec{u}]) = 1] = \mathrm{negl}(\lambda) \ ,$$

where the probability is taken over the output $\mathcal{MG}_k = (k, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_k(1^\lambda)$, $\mathbf{A} \leftarrow \mathcal{D}_{n,\ell}$, $\vec{w} \leftarrow \mathbb{Z}_p^\ell$, $\vec{u} \leftarrow \mathbb{Z}_p^n$ and the coin tosses of the adversary D.

We note that all of the standard examples of matrix assumptions are polynomially induced and further, in all examples we consider in this paper, the degree of $\mathfrak{p}_{i,j}$ is 1. In particular, we will refer to the following examples of matrix distributions, all for $n = \ell + 1$:

$$\mathcal{SC}_\ell : \mathbf{A} = \begin{pmatrix} -s & 0 & \ldots & 0 & 0 \\ 1 & -s & \ldots & 0 & 0 \\ 0 & 1 & & 0 & 0 \\ \vdots & & \ddots & \ddots & \\ 0 & 0 & \ldots & 1 & -s \\ 0 & 0 & \ldots & 0 & 1 \end{pmatrix} \quad \mathcal{L}_\ell : \mathbf{A} = \begin{pmatrix} s_1 & 0 & 0 & \ldots & 0 \\ 0 & s_2 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & 0 & \ldots & s_\ell \\ 1 & 1 & 1 & \ldots & 1 \end{pmatrix}, \quad \mathcal{U}_\ell : \mathbf{A} = \begin{pmatrix} s_{1,1} & \ldots & s_{1,\ell} \\ \vdots & & \vdots \\ s_{\ell+1,1} & \ldots & s_{\ell+1,\ell} \end{pmatrix},$$

where $s, s_i, s_{i,j} \leftarrow \mathbb{Z}_p$. Up to sign, the $\mathcal{SC}_\ell$ assumption, introduced in [9], is the *ℓ-symmetric cascade assumption* (ℓ-SCasc). The $\mathcal{L}_\ell$ assumption is actually the well-known *ℓ-linear assumption* (ℓ-Lin) [2, 15, 23] in matrix language (DDH equals 1-Lin), and the $\mathcal{U}_\ell$ assumption is the *ℓ-uniform assumption*. More generally, we can also define the $\mathcal{U}_{n,\ell}$ assumption for arbitrary $n > \ell$. Note that the $\mathcal{U}_{n,\ell}$ assumption is the weakest matrix assumption (with the worst representation size) and implied by any other $\mathcal{D}_{n,\ell}$ assumption [9]. In particular ℓ-Lin implies the ℓ-uniform assumption as shown by Freeman. Moreover, ℓ-SCasc, ℓ-Lin, and the ℓ-uniform assumption hold in the generic group model [24] relative to a k-linear group generator provided that $k \leq \ell$ [9].

Interpolating sets. Let $\vec{X} = (X_1, \ldots, X_d)$ be a vector of variables. Let $W \subset \mathbb{Z}_p[\vec{X}]$ be a subspace of polynomials of finite dimension m. Given a set of polynomials $\{\mathfrak{r}_0, \ldots, \mathfrak{r}_{m-1}\}$ which are a basis of W, we say that $\vec{x}_1, \ldots, \vec{x}_m \in \mathbb{Z}_p^d$ is an *interpolating set* for W if the matrix

$$\begin{pmatrix} \mathfrak{r}_0(\vec{x}_1) & \ldots & \mathfrak{r}_{m-1}(\vec{x}_1) \\ \vdots & & \vdots \\ \mathfrak{r}_0(\vec{x}_m) & \ldots & \mathfrak{r}_{m-1}(\vec{x}_m) \end{pmatrix}$$

has full rank. It can be easily seen that the property of being an interpolating set is independent of the basis. Further, when p is exponential (and m and the degrees of $\mathfrak{r}_i$ are polynomial) in the security parameter, any m random vectors $\vec{x}_1, \ldots, \vec{x}_m$ form an interpolating set with overwhelming probability.

3 Our Framework

We now present our definitional framework for composite-to-prime-order transformations. Basically, the definitions in this section will enable us to describe how groups of prime order p with a multilinear map e can be converted into groups of order p^n for some $n \in \mathbb{N}$ with a multilinear map $\tilde{e}$. These converted groups will then "mimic" certain features of composite-order groups. Since $\tilde{e}$ is just a composition of several instances of e, we will refer to e as the *basic multilinear map*. We start with an overview of the framework of Freeman ([10]), since this is the established model for such transformations. Afterwards, we describe our framework in terms of differences to the model of Freeman.

Freeman's model. Freeman identifies some abstract properties of bilinear composite order groups which are essential to construct some cryptographic protocols, namely subgroup indistinguishability, the projecting property and the canceling property. For Freeman, a symmetric bilinear map generator takes a bilinear group of prime order p with a pairing e and outputs some groups $\mathbb{H} \subset \mathbb{G}, \mathbb{G}_T$ of order p^n for some $n \in \mathbb{N}$ and a symmetric bilinear map $\tilde{e}\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, computed via the basic pairing e. Useful instances of such generators satisfy the subgroup indistinguishability assumption, which means that it should be hard to decide membership in $\mathbb{H} \subset \mathbb{G}$. Further, the pairing is projecting if the bilinear map generator also outputs some maps π, π_T defined respectively on $\mathbb{G}, \mathbb{G}_T$ which commute with the pairing and such that $\ker \pi = \mathbb{H}$. The pairing is canceling if $\tilde{e}(\mathbb{H}, \mathbb{H}') = 0$ for some decomposition $\mathbb{G} = \mathbb{H} \oplus \mathbb{H}'$.

Instantiations. Further, Freeman gives several concrete instantiations in which the subgroups $\mathbb{H}$ output by the generator are sampled uniformly. More specifically, in the language of [9], the instantiations sample subgroups according to the $\mathcal{U}_{n,\ell}$ distribution. Although his model is not specifically restricted to this case, follow-up work seems to identify "Freeman's model" with this specific matrix distribution. For instance, the results of [20] on the impossibility of achieving the projecting and canceling property simultaneously or the impossibility result of Seo [21], who proves a lower bound on the size of the image of a projecting pairing, are also in this setting.

Our model. Essentially, we recover Freeman's original definitions for the symmetric setting, however with some subtle additional precisions. First, we extend his model to multilinear maps and, like Seo [21], distinguish between basic multilinear map operations (e) and multilinear map operations ($\tilde{e}$), since an important efficiency measure is how many e operations are required to compute $\tilde{e}$. The second and main block of differences is introduced with the goal of making the model compatible with several families of matrix assumptions, yielding a useful tool to prove optimality and impossibility results. For this, we extend Freeman's model to explicitly support different families of subgroup assumptions and state clearly what the dependency relations between the different outputs of the multilinear group generator are. In Sec. 6 we explicitly discuss the advantages of the refinement of the model.

Definition 3. *Let $k, \ell, n, r \in \mathbb{N}$ with $k > 1$ and $r \geq n > \ell$. A $(k, (r, n, \ell))$ symmetric multilinear map generator $\mathcal{G}_{k,(r,n,\ell)}$ takes as input a security parameter 1^λ and a basic k-linear map generator $\mathcal{G}_k$ and outputs in probabilistic polynomial time a tuple $(\mathcal{MG}_k, \mathbb{H}, \mathbb{G}, \mathbb{G}_T, \tilde{e})$, where*

- *$\mathcal{MG}_k := (k, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_k(1^\lambda)$ is a description of a prime order symmetric k-linear group*
- *$\mathbb{G} \subset G^r$ is a subgroup of G^r with a minimal generating set of size n*
- *$\mathbb{H} \subset \mathbb{G}$ is a subgroup of $\mathbb{G}$ with a minimal generating set of size ℓ*
- *$\tilde{e}\colon \mathbb{G}^k \to \mathbb{G}_T$ is a non-degenerate k-linear map.*

We assume that elements in $\mathbb{H}, \mathbb{G}$ are represented as vectors in G^r. With this representation, it is natural to identify elements in these groups with vectors in $\mathbb{Z}_p^r$ in the usual way, via the canonical basis. Via this identification, any subgroup $\mathbb{H} \subset G^r$ spanned by $[\vec{b}_1], \ldots, [\vec{b}_\ell]$ corresponds to the subspace H of $\mathbb{Z}_p^r$ spanned

by $\vec{b}_1, \ldots, \vec{b}_\ell$, and we write $\mathbb{H} = [H]$. Further, we may assume that $\mathbb{G}_T = G_T^m$ and elements of $\mathbb{G}_T$ are represented by m-tuples of G_T, for some fixed $m \in \mathbb{N}$, although we do not include m as a parameter of the multilinear generator.

In most constructions $n = r$, in which case we drop the index r from the definition, and we simply refer to such a generator as a $(k, (n, \ell))$ generator $\mathcal{G}_{k,(n,\ell)}$. We always assume that membership in $\mathbb{G}$ is easy to decide.[2] In the case where $n = r$ and $\mathbb{G} = G^r$ this is obviously the case, but otherwise we assume that the description of $\mathbb{G}$ includes some auxiliary information which allows to test it (like in [22], [19] and our construction of Sec. B.2).

Definition 4 (Properties of multilinear map generators)**.** *Let $\mathcal{G}_{k,(r,n,\ell)}$ be a $(k, (r, n, \ell))$ symmetric multilinear map generator as in Def. 3 with output $(\mathcal{MG}_k, \mathbb{H}, \mathbb{G}, \mathbb{G}_T, \tilde{e})$. We define the following properties:*

- ***Subgroup indistinguishability.*** *We say that $\mathcal{G}_{k,(r,n,\ell)}$ satisfies the* subgroup indistinguishability property *if for all PPT adversaries D,*

$$\mathbf{Adv}_{\mathcal{G}_{k,(r,n,\ell)}}(\mathsf{D}) = \mathbf{Pr}[\mathsf{D}(\mathcal{MG}_k, \mathbb{H}, \mathbb{G}, \mathbb{G}_T, \tilde{e}, x) = 1] - \mathbf{Pr}[\mathsf{D}(\mathcal{MG}_k, \mathbb{H}, \mathbb{G}, \mathbb{G}_T, \tilde{e}, u) = 1] = \mathrm{negl}(\lambda) \ ,$$

  where the probability is taken over $(\mathcal{MG}_k, \mathbb{H}, \mathbb{G}, \mathbb{G}_T, \tilde{e}) \leftarrow \mathcal{G}_{k,(r,n,\ell)}(1^\lambda)$, $x \leftarrow \mathbb{H}, u \leftarrow \mathbb{G}$ and the coin tosses of the adversary D.
- ***Projecting.*** *We say that $(\mathcal{MG}_k, \mathbb{H}, \mathbb{G}, \mathbb{G}_T, \tilde{e})$ is* projecting *if there exist two non-zero homomorphisms $\pi\colon \mathbb{G} \to \mathbb{G}$, $\pi_T\colon \mathbb{G}_T \to \mathbb{G}_T$ such that $\ker \pi = \mathbb{H}$ and $\pi_T(\tilde{e}(x_1, \ldots, x_k)) = \tilde{e}(\pi(x_1), \ldots, \pi(x_k))$ for any $(x_1, \ldots, x_k) \in \mathbb{G}^k$. For the special case $r = n = \ell + 1$, $\mathbb{G} := G^n$ we can equivalently define the maps $\pi\colon G^n \to G$, $\pi_T\colon \mathbb{G}_T \to G_T$ such that $\ker \pi = \mathbb{H}$ and $\pi_T(\tilde{e}(x_1, \ldots, x_k)) = e(\pi(x_1), \ldots, \pi(x_k))$ (matching the original definition of [13]). As usual, we say that $\mathcal{G}_{k,(r,n,\ell)}$ is projecting if its output is projecting with overwhelming probability.*
- ***Canceling.*** *We say that $(\mathcal{MG}_k, \mathbb{H}_1, \mathbb{G}, \mathbb{G}_T, \tilde{e})$ is* canceling *if there exists a decomposition $\mathbb{G} = \mathbb{H}_1 \oplus \mathbb{H}_2$ such that for any $x_1 \in \mathbb{H}_{j_1}, \ldots, x_k \in \mathbb{H}_{j_k}$, $\tilde{e}(x_1, \ldots, x_k) = 0$ except for $j_1 = \ldots = j_k$. We call $\mathcal{G}_{k,(r,n,\ell)}$ canceling if its output is canceling with overwhelming probability.*

So far, the definitions given match those of Freeman (extended to the k-linear case) except that we explicitly define the basic k-linear group $\mathcal{MG}_k$ which is used in the construction. We will now introduce two aspects of our framework that are new compared to Freeman's model. First, we will define multilinear generators that sample subgroups according to a specific matrix assumptions. Then, we will define a property of the multilinear map $\tilde{e}$ that will be very useful to establish impossibility results and lower bounds.

Definition 5. *Let $k, \ell, n, r \in \mathbb{N}$ with $k > 1$, $r \geq n > \ell$ and $\mathcal{D}_{n,\ell}$ be a matrix distribution. A $(k, (r, n, \ell), \mathcal{D}_{n,\ell})$ multilinear map generator $\mathcal{G}_{k,(r,n,\ell),\mathcal{D}_{n,\ell}}$ is a $(k, (r, n, \ell))$ multilinear map generator which outputs a tuple $(\mathcal{MG}_k, \mathbb{H}, \mathbb{G}, \mathbb{G}_T, \tilde{e})$ such that the distribution of the subspaces H such that $\mathbb{H} = [H]$ equals $\mathcal{D}_{n,\ell}$ for any fixed choice of $\mathcal{MG}_k$.*

As usual, in the case where $r = n$, we just drop r and refer to a $(k, \mathcal{D}_{n,\ell})$ multilinear map generator $\mathcal{G}_{k,\mathcal{D}_{n,\ell}}$. We conclude our framework with a definition that enables us to distinguish generators where the multilinear map $\tilde{e}$ may or may not depend on the choice of the subgroups.

Definition 6. *We say that a $(k, (r, n, \ell), \mathcal{D}_{n,\ell})$ multilinear map generator with output $(\mathcal{MG}_k, \mathbb{H}, \mathbb{G}, \mathbb{G}_T, \tilde{e})$ as in Def. 5 defines a* fixed multilinear map *if the random variable H (s.t. $\mathbb{H} = [H]$) conditioned on $\mathcal{MG}_k$ and the random variable $(\mathbb{G}, \mathbb{G}_T, \tilde{e})$ conditioned on $\mathcal{MG}_k$ are independent.*

4 Our Constructions

All of our constructions arise from the following *polynomial point of view*: The key idea is to treat $\mathbb{G} = G^n$ as an implicit representation of some space of polynomials. Polynomial multiplication will then give us a

[2] We note that with the recent approximate multilinear maps from [11, 6], not even group membership is efficiently recognizable. This will not affect our results, but of course hinders certain applications (such as Groth-Sahai proofs).

Table 1: Efficiency of different symmetric projecting k-linear maps. The size of the domain (n) and codomain (m) of $\tilde{e}$ is given as number of group elements of G and G_T, respectively. Costs are stated in terms of application of the basic map e, group operations (gop) including inversion in G/G_T, and ℓ-fold multi-exponentiations of the form $e_1[a_1]+\cdots+e_\ell[a_\ell]$ (ℓ-mexp) in G/G_T. Note that in this paper, for the computation of $\tilde{e}$, we use an evaluate-multiply-approach.

Construction	**Ass.**	**Co-/Domain**	**Cost $\tilde{e}$**	**Cost π**	**Cost π_T**
Freeman, $k=2$ [10]	$\mathcal{U}_2$	9/3	9 e	3 3-mexp	9 9-mexp
Seo, $k=2$ [21]	$\mathcal{U}_2$	6/3	9 e + 3 gop	3 3-mexp	6 6-mexp
This paper, $k=2$	$\mathcal{SC}_2$	**5/3**	**5 e + 22 gop**	**1 2-mexp**	**1 5-mexp**
This paper, $k=2$	$\mathcal{U}_2$	6/3	6 e + 12 3-mexp[1]	1 3-mexp	1 6-mexp
Freeman, $k>2$	$\mathcal{U}_k$	$(k+1)^k/k+1$	$(k+1)^{k+1}$ e	$k+1$ $(k+1)$-mexp	$(k+1)^k$ $(k+1)^k$-mexp
This paper, $k>2$	$\mathcal{U}_k$	$\binom{2k}{k}/k+1$	$\binom{2k}{k}$ e + $\binom{2k}{k}k$ $(k+1)$-mexp[1]	1 $(k+1)$-mexp	1 $\binom{2k}{k}$-mexp
This paper, $k>2$	$\mathcal{SC}_k$	$k^2+1/k+1$	(k^2+1) e + (k^3+k) **k-mexp**[1]	**1 k-mexp**	**1 k^2+1-mexp**

[1]For the construction based on $\mathcal{SC}_k$, the involved exponents are relatively small, namely the biggest one is $(\lceil\frac{k^2+1}{2}\rceil)^k$. Also for $\mathcal{U}_k$, the involved exponents can usually be made small.

natural multilinear map. For subspaces $\mathbb{H}^{(\vec{s})}$ that correspond to polynomials sharing a common root $\vec{s}$, this multilinear map will turn out to be projecting. We will first illustrate this idea by means of a simple concrete example where subgroup decision for $\mathbb{H}^{(\vec{s})}$ is equivalent to 2-SCasc (Sec. 4.1). Then we show that actually any polynomially induced matrix assumption gives rise to such a polynomial space and thus allows for the construction of a k-linear projecting map (Sec. 4.2). Finally, by considering $\mathbb{G}$ along with the multilinear map as an implicit representation of a polynomial ring modulo some reducible polynomial, we are able to construct a multilinear map which is both projecting and canceling (see Sec. 4.3 for a summary). See Tab. 1 for an overview of the characteristics of our projecting map constructions in comparison with previous work.

4.1 A Projecting Pairing based on the 2-SCasc Assumption

Let $(k=2, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_2(1^\lambda)$ be the output of a symmetric prime-order bilinear group generator. We set $\mathbb{G} := G^3$ and $\mathbb{G}_T := G_T^5$. For any $[\vec{f}] = ([f_0],[f_1],[f_2]) \in \mathbb{G} = G^3$, we identify $\vec{f}$ with the polynomial $f = f_0 + f_1 X + f_2 X^2 \in \mathbb{Z}_p[X]$ of degree at most 2. Similarly, any $[\vec{f}]_T \in \mathbb{G}_T$ corresponds to a polynomial of degree at most 4. Then the canonical group operation for $\mathbb{G}$ and $\mathbb{G}_T$ corresponds to polynomial addition (in the exponent), i.e., $[\vec{f}]+[\vec{g}] = [\vec{f}+\vec{g}] = [f+g]$ and $[\vec{f}]_T + [\vec{g}]_T = [f+g]_T$. Furthermore, polynomial multiplication (in the exponent) gives a map $\tilde{e}\colon \mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$,

$$\tilde{e}([\vec{f}],[\vec{g}]) := \Big(\Big[\sum_{i+j=0} f_i g_j\Big]_T, \ldots, \Big[\sum_{i+j=4} f_i g_j\Big]_T\Big) = [f\cdot g]_T$$

It is easy to see that $(\mathbb{G}, \mathbb{G}_T, \tilde{e})$ is again a bilinear group setting, where the group operations and the pairing $\tilde{e}$ can be efficiently computed.

A subgroup decision problem. For some fixed $s \in \mathbb{Z}_p$ let us consider the subgroup $\mathbb{H}^{(s)} \subset \mathbb{G}$ formed by all elements $[\vec{f}] \in \mathbb{G}$ such that $\vec{f}$ viewed as polynomial f has root s, i.e., $\mathbb{H}^{(s)} = \{[f] \in \mathbb{G} \mid f(s) = 0\}$. In other words, $\mathbb{H}^{(s)}$ consists of all $[f]$ with f of the form

$$(X-s)(f_1' X + f_0') \ , \tag{1}$$

where $f_1', f_0' \in \mathbb{Z}_p$. Thus, given $[f]$ and $[s]$, the subgroup decision problem for $\mathbb{H}^{(s)} \subset \mathbb{G}$ means to decide whether f is of this form or not. Viewing Eq. (1) as matrix-vector multiplication, we see that this is equivalent to deciding whether $\vec{f}$ belongs to the image of the 3×2 matrix

$$\mathbf{A}(s) := \begin{pmatrix} -s & 0 \\ 1 & -s \\ 0 & 1 \end{pmatrix} \tag{2}$$

Hence, our subgroup decision problem corresponds to the 2-SCasc problem (cf. Def. 2) which is hard in a generic bilinear group [9].

Projections. Given s, we can simply define projection maps $\pi\colon \mathbb{G} \to G$ and $\pi_T\colon \mathbb{G}_T \to G_T$ by polynomial evaluation at s (in the exponent), i.e., $[\vec{f}]$ is mapped to $[f(s)]$ and $[\vec{f}]_T$ to $[f(s)]_T$. Computing π, π_T requires group operations only. Obviously, it holds that $\ker(\pi) = \mathbb{H}^{(s)}$ and $e(\pi([\vec{f}_1]), \pi([\vec{f}_2])) = \pi_T(\tilde{e}([\vec{f}_1], [\vec{f}_2]))$.

Sampling from $\mathbb{H}^{(s)}$. Given $[(-s, 1, 0)], [(0, -s, 1)] \in \mathbb{G}$, a uniform element from $\mathbb{H}^{(s)}$ can be sampled by picking $(f'_0, f'_1) \leftarrow \mathbb{Z}_p^2$ and, as with any matrix assumption, computing the matrix-vector product

$$\left[\begin{pmatrix} -s & 0 \\ 1 & -s \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} f'_0 \\ f'_1 \end{pmatrix}\right] = \left[(-sf'_0, f'_0 - sf'_1, f'_1)^{\mathrm{T}}\right] \tag{3}$$

Again, this can be done using the group operation only.

Efficiency. Computing $\tilde{e}$ in our construction corresponds to polynomial multiplication. Although this multiplication happens in the exponent (and we are "only" given implicit representations of the polynomials), we are not forced to stick to schoolbook multiplication. We propose to follow an evaluation-multiplication-interpolation approach (using small interpolation points) where the actual interpolation step is postponed to the computation of π_T.

More precisely, so far we used coefficient representation for polynomials over $\mathbb{G}$ and $\mathbb{G}_T$ with respect to the standard basis. However, other (s-independent) bases are also possible without affecting security. For efficiency, we propose to stick to this representation for $\mathbb{G}$ but to use point-value representation for polynomials over $\mathbb{G}_T$ with respect to the fixed interpolating set $M := \{-2, -1, 0, 1, 2\}$ (cf. Def. 2). This means we now identify a polynomial g in the target space with the vector $(g(-2), g(-1), g(0), g(1), g(2))$.

More concretely, to compute $\tilde{e}([f_1], [f_2]) = ([(f_1 f_2)(x)]_T)_{x \in M}$, we first evaluate f_1 and f_2 (in the exponent) with all $x \in M$, followed by a point-wise multiplication $([f_1(x) f_2(x)]_T)_{x \in M} = (e([f_1(x)], [f_2(x)]))_{x \in M}$. This way, $\tilde{e}$ can be computed more efficiently with only five pairings. Computing π is unchanged. To apply π_T, one first needs to obtain the coefficient representation by interpolation and then evaluate the polynomial at s. However, this can be done simultaneously and as the 1×5 matrix describing this operation can be precomputed (given s) it does not increase the computational cost much.

4.2 Projecting Multilinear Maps from any Matrix Assumption

In the following, we will first demonstrate that for any vector space of polynomials, the natural pairing given by polynomial multiplication is projecting for subspaces consisting of polynomials sharing a common root. We will then show that any (polynomially induced) matrix assumption can equivalently be considered as a subspace assumption in a vector space of polynomials of this type. This way, we obtain a natural projecting multilinear map for any polynomially induced matrix assumption.

A projecting multilinear map on spaces of polynomials. Let $\mathcal{MG}_k := (k, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_k(1^\lambda)$ be the output of a prime-order k-linear group generator. Let $V \subset \mathbb{Z}_p[\vec{X}]$ be a vector space of polynomials of dimension n for which we fix basis $\mathfrak{q}_0, \ldots, \mathfrak{q}_{n-1}$. Then for any $[\vec{f}] \in \mathbb{G} := G^n$ we can identify the vector $\vec{f} = (f_0, \ldots, f_{n-1})$ with a polynomial $f = \sum f_i \mathfrak{q}_i \in V$. In the 2-SCasc example above, V corresponds to univariate polynomials of degree at most 2 and the basis is given by $1, X, X^2$. On V, we have a natural k-linear map given by polynomial multiplication: $\mathrm{mult}_k\colon V^k \to \mathbb{Z}_p[\vec{X}], \mathrm{mult}_k(f_1, \ldots, f_k) = f_1 \cdots f_k$. Let $W \subset \mathbb{Z}_p[\vec{X}]$ be the span of the image of mult_k and m its dimension. Then we can again fix a basis $\mathfrak{r}_0, \ldots, \mathfrak{r}_{m-1}$ of W to identify polynomials with vectors. In the 2-SCasc example above, W consists of polynomials of degree at most 4 and we chose the basis $1, X, X^2, X^3, X^4$ of W for our initial presentation. From polynomial multiplication, we then obtain a non-degenerate k-linear map

$$\tilde{e}\colon \mathbb{G}^k \to G_T^m, \tilde{e}([\vec{f}_1], \ldots, [\vec{f}_k]) = [f_1 \cdots f_k]_T \ .$$

Now consider a subspace $\mathbb{H}^{(\vec{s})} \in \mathbb{G}$ of the form $\mathbb{H}^{(\vec{s})} = \{[f] \in \mathbb{G} \mid f(\vec{s}) = 0\}$. It is easy to see that $\tilde{e}$ is projecting for this subspace: A projection map $\pi\colon \mathbb{G} \to G$ with $\ker(\pi) = \mathbb{H}^{(\vec{s})}$ is given by evaluation at $\vec{s}$, i.e., $\pi([\vec{f}]) = [f(\vec{s})]$. Similarly, $\pi_T\colon G_T^m \to G_T$ is defined by $\pi_T([\vec{g}]_T) = [g(\vec{s})]_T$ and by construction we have $e(\pi([\vec{f}_1]), \ldots, \pi([\vec{f}_k])) = [f_1(\vec{s}) \cdots f_k(\vec{s})]_T = [(f_1 \cdots f_k)(\vec{s})]_T = \pi_T(\tilde{e}([\vec{f}_1], \ldots, [\vec{f}_k]))$.

From a polynomially induced matrix distribution to a space of polynomials. Now, let $\mathcal{D}_{n-1}$ be any polynomially induced matrix distribution as defined in Def. 2 and let $\mathbf{A}(\vec{X}) \in (\mathbb{Z}_p[\vec{X}])^{n\times(n-1)}$ be the polynomial matrix describing this distribution. Then we set $\mathbb{G} := G^n$ and consider the subspace $[\mathsf{Im}\,\mathbf{A}(\vec{s})]$ for some $\vec{s}$. We now show that we can identify $\mathbb{G}$ with a vector space V of polynomials, such that the subspace $\mathsf{Im}\,\mathbf{A}(\vec{s})$ corresponds exactly to polynomials having a root at $\vec{s}$. To this end, consider the determinant of $(\mathbf{A}(\vec{X})||\vec{F})$ as a polynomial $\mathfrak{d}$ in indeterminates $\vec{X}$ and $\vec{F}$. Since we assume that $\mathbf{A}(\vec{s})$ has generically[3] full rank a given vector $\vec{f} \in \mathbb{Z}_p^n$ belongs to the image of $\mathbf{A}(\vec{s})$ iff the determinant of the extended matrix $(\mathbf{A}(\vec{s})||\vec{f})$ is zero, i.e., $\mathfrak{d}(\vec{s},\vec{f}) = 0$. To obtain the desired vector space V with basis $\mathfrak{q}_0,\ldots,\mathfrak{q}_{n-1}$, we consider the Laplace expansion of this determinant to write $\mathfrak{d}$ as

$$\mathfrak{d}(\vec{X},\vec{F}) = \sum_{i=0}^{n-1} F_i \mathfrak{q}_i(\vec{X})\,. \tag{4}$$

for some polynomials $\mathfrak{q}_i(\vec{X})$ depending only on $\mathbf{A}$. For $\mathcal{SC}_2$, we have $\mathfrak{q}_i = X^i$. We note that in all cases of interest the $\mathfrak{q}_i$ are linearly independent (see [14]).

Thus, we may now identify $[\vec{f}] \in \mathbb{G}$ with the implicit representation of the polynomial $f = \mathfrak{d}(\vec{X},\vec{f}) = \sum_i f_i\mathfrak{q}_i$ and as $f(\vec{s}) = \sum_i f_i\mathfrak{q}_i(\vec{s}) = 0$ iff $\vec{f} \in \mathsf{Im}\,\mathbf{A}(\vec{s})$ we have $\mathbb{H}^{(\vec{s})} = [\mathsf{Im}\,\mathbf{A}(\vec{s})] = \{[f] \in \mathbb{G} \mid f(s) = 0\}$. Hence, we may construct a projecting k-linear map from polynomial multiplication as described in the previous paragraph.

Working through the construction, one can obtain explicit coordinates as follows: let W be the span of $\{\mathfrak{q}_{i_1}\cdots\mathfrak{q}_{i_k} \mid 0 \le i_j < n\}$ and fix a basis $\mathfrak{r}_0,\ldots,\mathfrak{r}_{m-1}$ of W. This determines coefficients $\lambda_t^{(i_1,\ldots,i_k)}$ in $\mathfrak{q}_{i_1}\cdots\mathfrak{q}_{i_k} = \sum_{t=0}^{m-1}\lambda_t^{(i_1,\ldots,i_k)}\mathfrak{r}_t$.

Recall that $\tilde{e}\colon (G^n)^k \to G_T^m$ is defined as $\tilde{e}([\vec{f_1}],\ldots,[\vec{f_k}]) = [f_1\cdots f_k]_T$, expressed as an element of G_T^m via the basis $\vec{\mathfrak{r}}$. In coordinates this reads

$$\tilde{e}([\vec{f_1}],\ldots,[\vec{f_k}]) = \Big(\sum_{j_1\le\ldots\le j_k}\lambda_0^{(j_1,\ldots,j_k)}\cdot\sum_{\substack{(i_1,\ldots,i_k)\in\\ \tau(j_1,\ldots,j_k)}} e([f_{1,i_1}],\ldots,[f_{k,i_k}]),\ldots, \tag{5}$$

$$\sum_{j_1\le\ldots\le j_k}\lambda_{m-1}^{(j_1,\ldots,j_k)}\cdot\sum_{\substack{(i_1,\ldots,i_k)\in\\ \tau(j_1,\ldots,j_k)}} e([f_{1,i_1}],\ldots,[f_{k,i_k}])\Big)$$

where $[f_{1,i_1}\cdots f_{k,i_k}]_T$ simply denotes $(f_{1,i_1}\cdots f_{k,i_k})\mathcal{P}_T$ and $\tau(j_1,\ldots,j_k)$ denotes the set of permutations of $(j_1,\ldots,j_k)$. The last optimization can be done as $\mathfrak{q}_{i_1}\cdots\mathfrak{q}_{i_k} = \mathfrak{q}_{j_1}\cdots\mathfrak{q}_{j_k}$ for $(i_1,\ldots,i_k) \in \tau(j_1,\ldots,j_k)$. For the same reason, we have $m = \binom{n+k-1}{k}$ in the worst case. In this way, the target group in our constructions is always smaller than the target group in Freeman's construction (generalized to $k \ge 2$), which is of size n^k.

The following theorem summarizes our construction and its properties:

Theorem 1. *Let $k > 1$, $n \in \mathbb{N}$, and $\mathcal{D}_{n-1}$ be a polynomially induced matrix distribution. Let $\mathcal{G}_{k,\mathcal{D}_{n-1}}$ be an algorithm that on input of a security parameter 1^λ and a symmetric prime-order k-multilinear map generator $\mathcal{G}_k$ outputs $(\mathcal{MG}_k, \mathbb{H}^{(\vec{s})}, \mathbb{G}, \mathbb{G}_T, \tilde{e})$, where*

- $\mathcal{MG}_k := (k, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_k(1^\lambda)$,
- $\mathbb{G} := G^n$, $\mathbb{H}^{(\vec{s})} := [\mathsf{Im}\,\mathbf{A}(\vec{s})]$, $\mathbf{A}(\vec{s}) \leftarrow \mathcal{D}_{n-1}$,
- $\mathbb{G}_T := G_T^m$, *where m equals the dimension of*

$$W := \{\sum_{0\le i_1,\ldots,i_k\le n-1}\alpha_{i_1,\ldots,i_k}\mathfrak{q}_{i_1}\cdots\mathfrak{q}_{i_k} \mid \alpha_{i_1,\ldots,i_k} \in \mathbb{Z}_p\}$$

[3] This means that $\mathbf{A}(\vec{s})$ will be full rank with overwhelming probability and this is indeed equivalent to $\mathfrak{d} \neq 0$. To simplify the exposition, we may assume that the sampling algorithm is changed to exclude $\vec{s}$ where $\mathbf{A}(\vec{s})$ does not have full rank.

(as vector space), and $\mathfrak{q}_0(\vec{X}), \ldots, \mathfrak{q}_{n-1}(\vec{X}) \in \mathbb{Z}_p[\vec{X}]$ *are polynomials such that*

$$\det(\mathbf{A}(\vec{X})||\vec{F}) = \sum_{i=0}^{n-1} F_i \mathfrak{q}_i(\vec{X})$$

for the matrix $\mathbf{A}(\vec{X})$ *describing* $\mathcal{D}_{n-1}$*, and*

- $\tilde{e}\colon \mathbb{G}^k \to \mathbb{G}_T$ *is the map defined by Eq. (5) for a basis* $\mathfrak{r}_0, \ldots, \mathfrak{r}_{m-1}$ *of* W*.*

Then $\mathcal{G}_{k,\mathcal{D}_{n-1}}$ *is a* $(k, \mathcal{D}_{n-1})$ *multilinear map generator. It is projecting, where the projection maps* $\pi\colon \mathbb{G} \to G$ *and* $\pi_T\colon \mathbb{G}_T \to G_T$ *defined by* $\pi(\vec{f}) := \sum_{i=0}^{n-1} \mathfrak{q}_i(\vec{s})[f_i]$ *and* $\pi_T(\vec{g}) := \sum_{i=0}^{m-1} \mathfrak{r}_i(\vec{s})[g_i]_T$ *are efficiently computable given the trapdoor* $\vec{s}$*. Furthermore, if the* $\mathcal{D}_{n-1}$ *assumption holds with respect to* $\mathcal{G}_k$*, then subgroup indistinguishability holds with respect to* $\mathcal{G}_{k,\mathcal{D}_{n-1}}$*.*

Example 1. *We can construct a projecting* k*-linear map generator satisfying subgroup indistinguishability under* k-SCasc *(which is hard in a* k*-linear generic group model). For* $\mathcal{G}_{k,\mathcal{SC}_k}$*, we would get* $n = k + 1$ *and* $\mathfrak{q}_i(X) = X^i$ *if* k *is even and* $\mathfrak{q}_i(X) = -X^i$ *when* k *is odd, where* $0 \le i \le k$*. Using the basis* $\mathfrak{r}_t(X) = X^t$ *for* W *if* k *is even and* $\mathfrak{r}_t(X) = -X^t$ *if* k *is odd for* $0 \le t \le k^2$*, we obtain* $\lambda_t^{(i_1,\ldots,i_k)} = 1$ *for* $t = i_1 + \cdots + i_k$ *and* $\lambda_t^{(i_1,\ldots,i_k)} = 0$ *else. Note that we have* $m = k^2 + 1$*.*

Example 2. *We can also construct a* k*-linear map generator from* k-Lin*. For* $\mathcal{G}_{k,\mathcal{L}_k}$*, we would have* $n = k+1$*, and polynomials* $\mathfrak{q}_k(X_0, \ldots, X_{k-1}) = X_0 \cdots X_{k-1}$ *and* $\mathfrak{q}_i(X_0, \ldots, X_{k-1}) = -\prod_{j \ne i} X_j$ *for* $0 \le i \le k-1$*. As a basis for* W *we can simply take* $\{\mathfrak{q}_{j_1} \cdots \mathfrak{q}_{j_k} \mid 0 \le j_1 \le \ldots \le j_k \le k\}$ *yielding* $m = \binom{n+k-1}{k}$*.*

Example 3. *Like Freeman, we could also construct a* k*-linear map generator from the* $\mathcal{U}_k$ *assumption. Although the polynomials* $\mathfrak{q}_i(X_{1,1}, \ldots, X_{k,k+1})$*,* $0 \le i \le k$*, associated to* $\mathcal{G}_{k,\mathcal{U}_k}$ *have a much more complex description than in the* k-Lin *case, the image size of the resulting map is the same, namely* $m = \binom{n+k-1}{k}$*, because a basis of the image is also* $\{\mathfrak{q}_{j_1} \cdots \mathfrak{q}_{j_k} \mid 0 \le j_1 \le \ldots \le j_k \le k\}$*.*

Efficiency. As in our setting any change of basis is efficiently computable, the security of our construction only depends on the vector space V (which in turn determines W), but not on the bases chosen. So we are free to choose bases that improve efficiency. We propose to follow the same approach as in Sec. 4.1: Select points $\vec{x}_0, \ldots, \vec{x}_{m-1}$ that form an interpolating set for W and represent $f \in W$ via the vector $f(\vec{x}_0), \ldots, f(\vec{x}_{m-1})$. This corresponds to choosing the basis of W consisting of polynomials $\mathfrak{r}_0, \ldots, \mathfrak{r}_{m-1} \in W$ such that $\mathfrak{r}_i(\vec{x}_j) = 1$ for $i = j$ and 0 otherwise. For the domain V, the choice is less significant and we might simply choose the $\mathfrak{q}_i$'s that the determinant polynomial gives us. Then we can compute $\tilde{e}([\vec{f}_1], \ldots, [\vec{f}_k])$ by an evaluate-multiply approach using only m applications of e. Note that the evaluation step can also be done pretty efficiently if the $\mathfrak{q}_i$'s have small coefficients (which usually is the case). For details see [14].

4.3 Canceling and Projecting k-Linear Maps From Polynomial Spaces

By considering polynomial multiplication modulo a polynomial h, which has a root at the secret s, we are able to construct a $(k, (n = \ell + 1, \ell))$ symmetric multilinear map generator with a *non-fixed* pairing that is both canceling and projecting. Our first construction relies on a $k' := k + 1$-linear prime-order map e. The one additional multiplication in the exponent is used to perform the reduction modulo h. Based on this construction, we propose another $(k, (r = 2\ell, n = \ell + 1, \ell))$ symmetric multilinear map generator that requires only a $k' = k$-linear prime-order map. The security of our constructions is based on variants of the ℓ-SCasc assumption. We need to extend ℓ-SCasc by additional given group elements to allow for reduction in the exponent, e.g., in the simplest case hints of the form $[X^i \bmod h]$ are given. We refer to Sec. B.1 and Sec. B.2 for details on the constructions and to Sec. C.2 for some efficiency considerations. In Sec. B.3 we show that our constructions are secure for $\ell \ge k'$ in generic k'-linear groups. We note that, to the best of our knowledge, this is the first construction of a projecting&canceling map that naturally generalizes to $k > 2$.

5 Optimality and Impossibility Results

5.1 Optimality of Polynomial Multiplication

In this section we show that for any polynomially induced matrix assumption $\mathcal{D}_{\ell+1,\ell}$, the projecting multilinear map resulting from the polynomial viewpoint is optimal in terms of image size.

Theorem 2. *Let $k > 0$, and let $\mathcal{D}_{\ell+1,\ell}$ be a polynomially induced matrix assumption and let $\mathfrak{q}_0, \ldots, \mathfrak{q}_\ell$ be the polynomials associated to $\mathcal{D}_{\ell+1,\ell}$ as defined in Eq. (4) in Sec. 4.2 and let $W \subset \mathbb{Z}_p[\vec{X}]$ be the space of polynomials spanned by $\{\mathfrak{q}_{i_1} \ldots \mathfrak{q}_{i_k} \mid 0 \leq i_j \leq \ell\}$. Let $(\mathcal{MG}_k, \mathbb{H}, G^{\ell+1}, G_T^m, \tilde{e})$ be the output of any other fixed $(k, \mathcal{D}_{\ell+1,\ell})$ projecting multilinear map generator. Then, $\overline{m} := \dim W \leq m$.*

$$\begin{array}{ccc} \mathbb{G}^k & \xrightarrow{\tilde{e}} & G_T^m \\ \downarrow{\scriptstyle (\pi^{(\vec{s})})^k} & & \downarrow{\scriptstyle \pi_T^{(\vec{s})}} \\ G^k & \xrightarrow{e} & G_T \end{array} \qquad \begin{array}{ccc} \mathbb{G}^k \times \ldots \times \mathbb{G}^k & \xrightarrow{(\tilde{e}, \ldots, \tilde{e})} & G_T^m \times \ldots \times G_T^m \\ \downarrow{\scriptstyle \left((\pi^{(\vec{s}_1)})^k, \ldots, (\pi^{(\vec{s}_{\overline{m}})})^k\right)} & & \downarrow{\scriptstyle \left(\pi_T^{(\vec{s}_1)}, \ldots, \pi_T^{(\vec{s}_{\overline{m}})}\right)} \\ G^k \times \ldots \times G^k & \xrightarrow{(e, \ldots, e)} & G_T \times \ldots \times G_T \end{array}$$

Figure 1: Left: Projecting property. Right: The diagram repeated $\overline{m}$ times for an interpolating set $\vec{s}_1, \ldots \vec{s}_{\overline{m}}$ for W.

Proof Intuition. The first part of the proof shows that w.l.o.g. we can assume that $\pi_T^{(\vec{s})} \circ \tilde{e}$ is polynomial multiplication for all $\vec{s}$, that is, for any $[\vec{f}_1], \ldots, [\vec{f}_k] \in G^{\ell+1}$, $\pi_T(\tilde{e}([\vec{f}_1], \ldots, [\vec{f}_k])) = [(f_1 \ldots f_k)(\vec{s})]_T$. This follows from the commutative diagram on the left, i.e. the projecting property, together with the fact that, because $\mathbb{H}$ has codimension 1, the map $\pi^{(\vec{s})}$ must (up to scalar multiples) correspond to polynomial evaluation at $\vec{s}$. The intuition for the second part of the proof is given by Fig. 1. Here we show that if $\vec{s}_1, \ldots \vec{s}_{\overline{m}}$ is an interpolating set for W, then the span of $\left\{ \left(\pi_T^{(\vec{s}_1)}(\vec{x}), \ldots, \pi_T^{(\vec{s}_{\overline{m}})}(\vec{x})\right) | \vec{x} \in \tilde{e}(\mathbb{G}^k) \right\} \subset G_T^{\overline{m}}$ is of dimension $\overline{m}$. This dimension can be at most the dimension of the span of $\tilde{e}(\mathbb{G}^k)$, showing $\overline{m} \leq m$. A full proof is given in [14].

5.2 Optimality of our Projecting Multilinear Map from the SCasc-Assumption

As a result of our general viewpoint, we can actually show that the projecting multilinear map based on the SCasc-assumption is optimal among *all* polynomially induced matrix assumptions $\mathcal{D}_{n,\ell}$ that are not redundant. Non-redundancy rules out the case where some components of $\vec{z}$ are no help (even information-theoretically) in distinguishing $\vec{z} \in \mathbb{G}$ from $\vec{z} \in \mathbb{H}^{(s)}$. See [14] for a formal definition.

Theorem 3. *Let $n = \ell + 1$ and $\mathcal{D}_{n,\ell}$ be a polynomially induced matrix distribution which is not redundant. Let $(\mathcal{MG}_k, \mathbb{H}, G^n, G_T^m, \tilde{e})$ be the output of some projecting $(k, \mathcal{D}_{n,\ell})$ multilinear map generator with a fixed multilinear map. Then, $m \geq \ell k + 1$.*

Note that the projecting pairing based on the polynomial viewpoint of the ℓ-SCasc-assumption reaches this bound and is hence optimal.

Proof. We may identify G^n with some subspace $V \subset \mathbb{Z}_p[\vec{X}]$ of dimension n (see [14] for details). By Thm. 2 above, we may assume w.l.o.g. that $\tilde{e}$ is polynomial multiplication, as this only makes m smaller. Hence we can also identify G_T^m with some subspace $W \subset \mathbb{Z}_p[\vec{X}]$ of dimension m. Let $>$ be any monomial ordering on $\mathbb{Z}_p[\vec{X}]$. Let $\mathfrak{q}_0, \ldots, \mathfrak{q}_\ell$ be a basis of V in echelon form with respect to $>$. This implies that the leading monomials satisfy $\mathrm{LM}(\mathfrak{q}_0) > \ldots > \mathrm{LM}(\mathfrak{q}_\ell)$. Now consider the elements
(the definition of $\mathfrak{r}_{i+1}$ differs from that of $\mathfrak{r}_i$ in one single index being greater by one). All $\mathfrak{r}_i \in W$ by construction and $\mathrm{LM}(\mathfrak{r}_0) > \mathrm{LM}(\mathfrak{r}_1) > \ldots > \mathrm{LM}(\mathfrak{r}_{\ell k})$ by the properties of a monomial order. It follows that the $\mathfrak{r}_i$ are linearly independent, showing $m = \dim W \geq \ell k + 1$. □

$$
\begin{array}{lll}
\mathfrak{q}_0^k = \mathfrak{r}_0 = \mathfrak{q}_0 \cdots \mathfrak{q}_0 \mathfrak{q}_0 & & \\
\mathfrak{r}_1 = \mathfrak{q}_0 \cdots \mathfrak{q}_1 \mathfrak{q}_0 & \mathfrak{r}_{\ell+1} = \mathfrak{q}_0 \cdots \mathfrak{q}_0 \mathfrak{q}_1 \mathfrak{q}_\ell & \cdots \quad \mathfrak{r}_{(k-1)\ell+1} = \mathfrak{q}_0 \mathfrak{q}_\ell \cdots \mathfrak{q}_\ell \\
\vdots & \vdots & \cdots \quad \vdots \\
\mathfrak{r}_\ell = \mathfrak{q}_0 \cdots \mathfrak{q}_0 \mathfrak{q}_\ell & \mathfrak{r}_{2\ell} = \mathfrak{q}_0 \cdots \mathfrak{q}_0 \mathfrak{q}_\ell \mathfrak{q}_\ell & \cdots \quad \mathfrak{r}_{\ell k} = \mathfrak{q}_\ell \mathfrak{q}_\ell \cdots \mathfrak{q}_\ell
\end{array}
$$

6 Review of Previous Results in our Framework

Let us consider some previous results using the language introduced in Sec. 3.

Projecting Pairings. Implicitly, in [13], Groth and Sahai were using the fact that the bilinear symmetric tensor product is a projecting map. Subsequently, Seo [21] constructed an improved symmetric projecting pairing which he claimed to be optimal in terms of image size and operations.

Theorem 4. *([21]) Let $\mathcal{G}_{2,\mathcal{U}_\ell}$ be any (symmetric) projecting $(2,\mathcal{U}_\ell)$ bilinear map generator with output $(\mathcal{MG}_2, \mathbb{H}, \mathbb{G}, G_T^m, \tilde{e})$. Then (a) we have $m \geq (\ell+1)(\ell+2)/2$, and (b) the map $\tilde{e}$ cannot be evaluated with less than $(\ell+1)^2$ prime-order pairing operations.*

Using the polynomial point of view, we prove in [14] that polynomial multiplication is optimal for *any* $\mathcal{D}_\ell$ assumption, and thus cover Thm. 4 (a) as a special case when $\mathcal{D}_\ell = \mathcal{U}_\ell$. On the other hand, the polynomial viewpoint immediately suggests a method to evaluate Seo's pairing with m (less than $(\ell+1)^2$) prime-order pairing operations, refuting Thm. 4 (b).[4] Further, our results also answer in the affirmative an open question raised by Seo about the existence of more efficient pairings outside of the model. Our construction of a k-linear map based on k-SCasc beats this lower bound and is much more efficient asymptotically in k.

Canceling and Projecting Pairings. In his original paper [10], Freeman gives several constructions of bilinear pairings which are either projecting or canceling — but not both. Subsequently, Meiklejohn *et al.* [20] give evidence that it might be hard to obtain both features simultaneously:

Theorem 5. *([20]) Any symmetric $(2,\mathcal{U}_\ell)$ bilinear generator with a fixed pairing cannot be simultaneously projecting and canceling, except with negligible probability (over the output of the generator).*[5]

In [14] we show that this result can be extended to any $(2,\mathcal{L}_\ell)$ and any $(2,\mathcal{SC}_2)$ bilinear generator. It remains an open question if the impossibility results extend to $(2,\mathcal{SC}_\ell)$, for $\ell > 2$.

With these impossibility results, it is not surprising that all canceling and projecting constructions are for *non-fixed* pairings in the sense of Def. 6. Indeed, in [22] Cheon and Seo construct a pairing which is both canceling and projecting but not fixed since, implicitly, the group $\mathbb{G}$ depends on the hidden subgroup $\mathbb{H}$. In our language, the pairing of Seo and Cheon is a $(2,(r=\ell^2, n=\ell+1, \ell))$ pairing, i.e., $\mathbb{G} \subset G^{\ell^2}$ of dimension $n = \ell+1$. Recently, Lewko and Meiklejohn [19] simplified this construction, obtaining a $(2,(r=2\ell, n=\ell+1, \ell))$ bilinear map generator. In [14] we also construct a $(2,(r=2\ell, n=\ell+1, \ell))$ pairing achieving both properties (and which generalizes to any $(k,(r=2\ell, n=\ell+1, \ell))$ with $\ell \geq k$) , but using completely different techniques. A direct comparison of [22], [19] with our pairing is not straightforward, since in fact they use dual vector spaces techniques and their pairing is not really symmetric.

7 A Direct Application: More Efficient Groth-Sahai Proofs

In this section, we will exemplarily illustrate how applications benefit from our more efficient and general constructions. Using our projecting pairing from Sec. 4.1, we can improve the performance of Groth-Sahai proofs by almost halving the number of required prime-order pairing operations (cf. Tab. 1). Additionally, in

[4] In [14] we discuss in more detail Seo's construction and the reason why Thm. 4 (b) is false.

[5] Their claim is that it is impossible to achieve both properties under what they call a "natural use" of the ℓ-Lin assumption although they are really using the uniform assumption.

Sec. F.2 in the Appendix, we show how to implement a k-linear variant of the Boneh-Goh-Nissim encryption scheme [3] using the projecting multilinear map generator $\mathcal{G}_{k,\mathcal{SC}_k}$.

Groth-Sahai proofs [13] are the most natural application of projecting bilinear maps. They admit various instantiations in the prime-order setting. It follows easily from the original formulation of Groth and Sahai that their proofs can be instantiated based on any $\mathcal{D}_{n,\ell}$ assumption and any fixed projecting map. Details are given in [9] but only for the projecting pairing corresponding to the symmetric bilinear tensor product. The generalization to any projecting pairing is straightforward, additional details are given in Sec. F.1.

The important parameters for efficiency of NIZK proofs are the size of the common reference string, the proof size and the verification cost. The proof size (for a given equation) depends only on the size of the matrix assumption, that is of n, ℓ, so it is omitted in our comparison. The size of the common reference string depends essentially on the size of the commitment key, which is $n + \mathrm{Re}_{\mathbb{G}}(\mathcal{D}_{n,\ell})$, where $\mathrm{Re}_{\mathbb{G}}(\mathcal{D}_{n,\ell})$ is the representation size of the matrix assumption $\mathcal{D}_{n,\ell}$, which is 1 for ℓ-SCasc, ℓ for ℓ-Lin and $(\ell+1)\ell$ for $\mathcal{U}_\ell$. Therefore, the ℓ-SCasc instantiation is the most advantageous from the point of view of the size of the common reference string (regardless of the pairing used), as pointed out in [9].

On the other hand, the choice of the pairing affects only the cost of verification. Except for some restricted type of linear equations, typically, verification involves several evaluations of $\tilde{e}$. In our most efficient construction, for each pairing evaluation $\tilde{e}$, we save, according to Tab. 1, at least 4 prime-order pairing evaluations. For instance, this leads to a saving of 12 pairing evaluations for proving that a committed value is a bit $b \in \{0, 1\}$.

Acknowledgements

We would like to thank the anonymous reviewers for very helpful and constructive comments. This work has been supported in part by DFG grant GZ HO 4534/4-1. Carla Ràfols was supported by a Sofja Kovalevskaja Award of the Alexander von Humboldt Foundation and the German Federal Ministry for Education and Research. The authors would like to thank Keita Xagawa for pointing out a minor mistake in the description of the 2-SCasc instantiation of GS Proofs of Sec. F.1.

References

[1] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer, May 2005. 22

[2] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, Aug. 2004. 4

[3] D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-DNF formulas on ciphertexts. In J. Kilian, editor, *TCC 2005: 2nd Theory of Cryptography Conference*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341. Springer, Feb. 2005. 1, 3, 13, 29, 30

[4] D. Boneh and B. Waters. A fully collusion resistant broadcast, trace, and revoke system. In A. Juels, R. N. Wright, and S. Vimercati, editors, *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 211–220. ACM Press, Oct. / Nov. 2006. 1

[5] X. Boyen. The uber-assumption family (invited talk). In S. D. Galbraith and K. G. Paterson, editors, *PAIRING 2008: 2nd International Conference on Pairing-based Cryptography*, volume 5209 of *Lecture Notes in Computer Science*, pages 39–56. Springer, Sept. 2008. 22

[6] J.-S. Coron, T. Lepoint, and M. Tibouchi. Practical multilinear maps over the integers. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 476–493. Springer, Aug. 2013. 4, 6, 33, 34

[7] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976. 1

[8] A. Escala and J. Groth. Fine-tuning Groth-Sahai proofs. In *PKC 2014: 17th International Workshop on Theory and Practice in Public Key Cryptography*, Lecture Notes in Computer Science, pages 630–649. Springer, 2014. 27, 29

[9] A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An algebraic framework for Diffie-Hellman assumptions. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 129–147. Springer, Aug. 2013. 2, 3, 4, 5, 7, 13, 20, 21, 22, 27, 28, 29

[10] D. M. Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In H. Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 44–61. Springer, May 2010. 1, 2, 5, 7, 12, 29, 30, 31

[11] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In T. Johansson and P. Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17. Springer, May 2013. 4, 6, 33, 34

[12] K. Gjøsteen. Symmetric subgroup membership problems. In S. Vaudenay, editor, *PKC 2005: 8th International Workshop on Theory and Practice in Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 104–119. Springer, Jan. 2005. 1

[13] J. Groth and A. Sahai. Efficient noninteractive proof systems for bilinear groups. *SIAM J. Comput.*, 41(5):1193–1232, 2012. 2, 6, 12, 13, 27, 28, 31

[14] G. Herold, J. Hesse, D. Hofheinz, C. Ràfols, and A. Rupp. Polynomial spaces: A new framework for composite-to-prime-order transformations. Cryptology ePrint Archive, 2014. `http://eprint.iacr.org/`. 9, 10, 11, 12

[15] D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. In A. Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 553–571. Springer, Aug. 2007. 4

[16] A. Karatsuba and Y. Ofman. Multiplication of many-digital numbers by automatic computers. In *USSR Academy of Sciences*, volume 145, pages 293–294, 1962. 23

[17] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In N. P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, Apr. 2008. 1

[18] A. B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 318–335. Springer, Apr. 2012. 2

[19] A. B. Lewko and S. Meiklejohn. A profitable sub-prime loan: Obtaining the advantages of composite-order in prime-order bilinear groups. *IACR Cryptology ePrint Archive*, 2013:300, 2013. 2, 6, 12

[20] S. Meiklejohn, H. Shacham, and D. M. Freeman. Limitations on transformations from composite-order to prime-order groups: The case of round-optimal blind signatures. In M. Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 519–538. Springer, Dec. 2010. 1, 2, 5, 12, 25

[21] J. H. Seo. On the (im)possibility of projecting property in prime-order setting. In X. Wang and K. Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 61–79. Springer, Dec. 2012. 2, 5, 7, 12, 29, 30, 31

[22] J. H. Seo and J. H. Cheon. Beyond the limitation of prime-order bilinear groups, and round optimal blind signatures. In R. Cramer, editor, *TCC 2012: 9th Theory of Cryptography Conference*, volume 7194 of *Lecture Notes in Computer Science*, pages 133–150. Springer, Mar. 2012. 2, 3, 6, 12

[23] H. Shacham. A Cramer-Shoup encryption scheme from the linear assumption and from progressively weaker linear variants. *IACR Cryptology ePrint Archive*, 2007:74, 2007. 4

[24] V. Shoup. Lower bounds for discrete logarithms and related problems. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer, May 1997. 4

[25] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In S. Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, Aug. 2009. 1, 2

[26] A. Weimerskirch and C. Paar. Generalizations of the Karatsuba algorithm for efficient implementations. Cryptology ePrint Archive, Report 2006/224, 2006. `http://eprint.iacr.org/`. 23

[27] A. Zanoni. Iterative Karatsuba method for multivariate polynomial multiplication. In D. W. Daniel Breaz, Nicoleta Breaz, editor, *Proceedings of the International Conference on Theory and Applications of Mathematics and Informatics, ICTAMI 2009*, pages 829–843. Aeternitas Publishing House, September 2009. 23

A The Polynomial Viewpoint

In the construction of our projecting multilinear in Sec. 4.2, we claimed that the $\mathfrak{q}_i$'s we obtained there were linearly independent for all interesting matrix assumptions. We will make this more precise now, saying what the uninteresting matrix assumptions are here: For this, let $\mathbf{A} \in (\mathbb{Z}_p[\vec{X}])^{n\times(n-1)}$ be a matrix describing a generically full rank, polynomially induced matrix assumption. This gives us subspaces $\mathbb{H}^{(\vec{s})} \subset G^n$ with $\mathbb{H}^{(s)} = [\mathsf{Im}\,\mathbf{A}(\vec{s})]$. Consider the case where for any fixed value of $\vec{s}$, for $\vec{w} \in \mathbb{Z}_p^n$ uniform, the distribution of one of the components, say the last one, of $\mathbf{A}(\vec{s})\vec{w}$ is uniform and independent from the other components. This last component then has no bearing whatsoever on the hardness of distinguishing $([\mathbf{A}(\vec{s})], [\mathbf{A}(\vec{s})\vec{w}])$ from $([\mathbf{A}(\vec{s})], [\vec{u}])$ for $[\vec{u}]$ uniform and we might just as well drop it. Slightly more generally, consider the following definition:

Definition 7. *Let $\mathbf{A} \in (\mathbb{Z}_p[\vec{X}])^{n\times(n-1)}$ be a matrix describing a (generically) full rank, polynomially induced matrix distribution as above. We call $\mathbf{A}$ or its associated matrix distribution redundant, if there exists a matrix $\mathbf{B} \in \mathbb{Z}_p^{n\times n}$, independent from $\vec{s}$, such that for all fixed $\vec{s}$ the last component of $\mathbf{B}\cdot\mathbf{A}(\vec{s})\vec{w}$ is uniform and independent from the other components over a uniformly random choice of $\vec{w}$.*

Even if the $\mathfrak{q}_i$'s are not linearly independent, we can still view elements as polynomials as follows: as in Sec. 4.2, consider the determinant polynomial $\mathfrak{d} = \det(\mathbf{A}(\vec{X})||\vec{F})$ as a polynomial in $\vec{X}, \vec{F}$ and let

$$\mathfrak{d} = \sum \mathfrak{q}_i(\vec{X}) \cdot F_i$$

be its Laplace expansion. So the $\mathfrak{q}_i$'s are (up to sign) the determinants of the $(n-1)\times(n-1)$-minors of $\mathbf{A}$. Let $V \subset \mathbb{Z}_p[\vec{X}]$ be their span. Even if the $\mathfrak{q}_i$'s may be linearly dependent, we can still map vectors to polynomials as we did before. For any $[\vec{f}] \in G^n$, we can consider the polynomial $\sum f_i\mathfrak{q}_i$. This means we have a surjective map

$$\Phi\colon\ G^n \cong \mathbb{G} \to V, \Phi([\vec{f}]) = \sum_i f_i\mathfrak{q}_i(\vec{X}) \tag{6}$$

realizing the polynomial viewpoint.

Theorem 6. *Let $\mathbf{A} \in (\mathbb{Z}_p[\vec{X}])^{n\times(n-1)}$ be a matrix describing a generically full rank, polynomially induced matrix distribution, which is not redundant. Then Φ is bijective.*

Proof. Consider the case where Φ is not injective. Then there exist a non-zero vector $[\vec{v}] \in \ker \Phi$. By definition, $\Phi([\vec{v}])(\vec{s}) = \sum_i \mathsf{q}_i(\vec{s})v_i = 0 = \mathfrak{d}(\vec{s},\vec{v})$ for all $\vec{s}$. So actually, $[\vec{v}] \in \mathbb{H}^{(\vec{s})} = [\mathsf{Im}\,\mathbf{A}(\vec{s})]$ for all $\vec{s}$. Let $\mathbf{B}$ be some invertible matrix such that $\mathbf{B}\vec{v}$ is the last unit vector. Then $(0,\ldots,0,1) \in \mathsf{Im}\,\mathbf{B}\mathbf{A}(\vec{s})$, hence the last component from a random element from this image is uniform and independent from the other components, contradicting that $\mathbf{A}$ is not redundant. □

We remark that, by setting $\tilde{e}([\vec{f_1}],\ldots,[\vec{f_k}]) := [\Phi(\vec{f_1})\cdots\Phi(\vec{f_k})]_T$, we can define a projecting multilinear map either way. If Φ is not injective, the effect of Φ is exactly to drop any redundant components. The only place where we need that Φ is injective is for the lower bounds in our optimality proof in Sec. 5.2. Of course, with redundant matrix assumptions, one can beat this lower bound as follows: Take a projecting multilinear map for a $\mathcal{D}_{n,n-1}$ matrix assumption with image size m and artificially increase n by redundant components (this corresponds to adding an identity matrix block with $\mathbf{A}$ becoming block diagonal) and have the multilinear map ignore them (which is what our map does).

B Projecting & Canceling Multilinear Maps from an Extended SCasc Assumption

B.1 First Construction Based on a $(k+1)$-linear Prime-Order Map

In order to obtain a multilinear map that is both projecting and canceling we modify our construction based on SCasc from Sec. 4.1. On a high level, our construction works as follows. We will again identify vectors from $\mathbb{G} = G^n$ with polynomials $\mathbb{Z}_p[X]$ (in the exponent) with polynomial addition as the group operation. But now, our k-linear map will correspond to polynomial multiplication *modulo some polynomial* $h(X)$ (where $h(X)$ will depend on s). To retain the projecting property, we ensure that $h(X)$ has a root at s, so $X-s$ divides $h(X)$. The orthogonal complement to $\mathbb{H}^{(s)}$ for the canceling property will then correspond to the span of $\frac{h(X)}{X-s}$, using the fact that $(X-s)\cdot\frac{h(X)}{X-s} \mod h = 0$

We want to point out that, since in this construction modular reduction consumes one multiplication in the exponent, to emulate a canceling k-linear map $\tilde{e}$, our first construction will require a $k' = k+1$-linear basic prime-order map e. This will be improved in the following in Sec. B.2.

Let $2 \le k < k' < n$.[6] We start with a basic symmetric k'-linear group generator $(k', G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_{k'}$ for groups of prime order p. The polynomial by which we reduce is chosen as follows: Fix any degree n polynomial $h'(X)$, which for efficiency we select as $h'(X) := X^n$ and set $h(X) = h'(X) - h'(s) = X^n - s^n$ for $s \xleftarrow{R} \mathbb{Z}_p$. This choice ensures that $X-s$ divides h. We set $\mathbb{G} := G^n$ and $\mathbb{G}_T := G_T^n$ and note that, with the notation from Sec. 4.1, we can identify (using coefficient representation for polynomials everywhere) these two sets with the ring $\mathbb{Z}_p[X]/(h)$, whose elements are represented by polynomials $f \in \mathbb{Z}_p[X]$ of degree at most $n-1$. We again use polynomial addition as group operation and define our composite-order k-linear map $\tilde{e} : \mathbb{G}\times\cdots\times\mathbb{G} \to \mathbb{G}_T$ by polynomial multiplication modulo the polynomial h:

$$\tilde{e}([\vec{f_1}],\ldots,[\vec{f_k}]) := [f_1\cdots f_k \bmod h]_T$$

This requires reducing a polynomial of degree $k(n-1)$ modulo h. To perform this, the crucial observation is that the map $g \mapsto g \bmod h$ sending a polynomial $g = \sum g_i X^i$ of degree at most $k(n-1)$ to a polynomial of degree at most $n-1$ is a *linear* map for h fixed. Viewed as a matrix, its coefficients are given by $h_{i,j}$ where $h_i(X) := X^i \bmod h = \sum_{j=0}^{n-1} h_{i,j}X^j$ for $0 \le i \le k(n-1)$. In other words, $g \bmod h = \sum_j \sum_{i=0}^{k(n-1)} g_i h_{i,j} X^j$.

[6] Our construction works for arbitrary $n > k'$. A larger n leads to a less efficient construction, but also permits a security proof based on a weaker assumption.

Combining this with the definition of polynomial multiplication, we thus may compute $\tilde{e}([f_1],\ldots,[f_k])$ as

$$\Big(\sum_{i=0}^{k(n-1)}\sum_{j_1+\ldots+j_k=i}[h_{i,0}\cdot f_{1,j_1}\cdots f_{k,j_k}]_T,\ldots,\sum_{i=0}^{k(n-1)}\sum_{j_1+\ldots+j_k=i}[h_{i,n-1}\cdot f_{1,j_1}\cdots f_{k,j_k}]_T\Big) \tag{7}$$

where for our choice of $h'(X)=X^n$, $h_{i,j}=0$ if $j\neq i \bmod n$ and $h_{j+\ell n,j}=s^\ell$. The k-linear pairing $\tilde{e}$ is efficiently[7] computable with the basic $k'\geq k+1$-linear map e, provided we know the $[h_{i,j}]$ that appear here. For this reason, we need to publish the $[h_{i,j}]$ as additional "hints". For our choice of $h'(X)$, this means publishing the $\kappa=k-1$ hints $[s^n],[s^{2n}],\ldots[s^{n(k-1)}]$. It is easy to see that, for other choices of (publicly known) $h'(X)$, all $[h_{i,j}]$ can be efficiently computed from $[h'(s)],\ldots,[h'(s)^{k-1}]$ as linear functions and vice versa. Of course, publishing these hints changes the security assumption we have to make. We will show in Thm. 7 that our construction is secure in the generic k'-linear group model.

For the smallest meaningful choice $n=4,k=2,k'=3$, our construction translates to

$$\tilde{e}([\vec{f}],[\vec{f'}])=\Big([f_0f'_0+s^4f_1f'_3+s^4f_2f'_2+s^4f_3f'_1]_T,[f_0f'_1+f_1f'_0+s^4f_2f'_3+s^4f_3f'_2]_T, \tag{8}$$

$$[f_0f'_2+f_1f'_1+f_2f'_0+s^4f_3f'_3]_T,[f_0f'_3+f_1f'_2+f_2f'_1+f_3f'_0]_T\Big) \tag{9}$$

which can be computed with a basic 3-linear map e from $[\vec{f}],[\vec{f'}],[s^4]$ using 16 evaluations of e.

Subgroups. Again, consider the subgroup $\mathbb{H}^{(s)}\subset\mathbb{G}$ formed by all elements $[\vec{f}]\in\mathbb{G}$ such that $f(s)=0$. Note that, since $X-s$ divides h, reducing modulo h does not change whether a polynomial has a root at s. As seen before, deciding membership in $\mathbb{H}^{(s)}$ is equivalent to deciding whether an element lies in the image of an $n\times(n-1)$-matrix $\mathbf{A}(s)$ from Eq. (2). Since the polynomials $[h_i]$ provide additional information about s, subgroup indistinguishability does not correspond to the $(n-1)$-SCasc assumption anymore, but to the new extended $(\kappa=k-1,n-1)$-SCasc assumption relative to $\mathcal{G}_{k'}$ defined below. We will discuss its generic security in Sec. B.3, together with our next construction.

Definition 8 (Extended SCasc assumption)**.** *Let $k',n,\kappa,\in\mathbb{N}$, $k'<n$ and consider $(k',G,G_T,e,p,\mathcal{P},\mathcal{P}_T)\leftarrow\mathcal{G}_{k'}$ for a k'-linear map generator $\mathcal{G}_{k'}$. Set $h(X):=X^n-s^n$ for $s\xleftarrow{R}\mathbb{Z}_p$. Let $\mathbf{A}\in\mathbb{Z}_p^{n\times n-1}$ be of the form (2) with $-s$ in the main diagonal and $\vec{w}\in\mathbb{Z}_p^{n-1}$, $\vec{u}\in\mathbb{Z}_p^n$. We say that the* extended $(\kappa,n-1)$-SCasc assumption *holds relative to $\mathcal{G}_{k'}$ if for all PPT algorithms $\mathcal{A}$ we have that*

$$\big|\mathbf{Pr}\big[\mathcal{A}([s^n],[s^{2n}],\ldots,[s^{\kappa n}],[\mathbf{A}],[\mathbf{A}\vec{w}])=1\big]-\mathbf{Pr}\big[\mathcal{A}([s^n],[s^{2n}]\ldots,[s^{\kappa n}],[\mathbf{A}],[\vec{u}]))=1\big]\big|$$

is negligible, where the probability is taken over the random choices of $s,h',\vec{w},\vec{u}$.

Projecting the elements of $\mathbb{H}^{(s)}$ to $0_\mathbb{G}$ and sampling from the subgroups works as in Sec. 4.1. This uses that $(f \bmod h)(s)=f(s)$ if $X-s$ divides h. Additionally, we have $\mathbb{G}=\mathbb{H}^{(s)}\oplus\mathbb{H}^{(s,\perp)}$, where $\mathbb{H}^{(s,\perp)}:=\{[f]\in\mathbb{G}\mid\exists\alpha\in\mathbb{Z}_p[X]:f(X)=\alpha\frac{h}{X-s}\}$. Note that $h(X)$ has no double root at s with overwhelming probability, so this sum is a direct sum. It holds that $\tilde{e}([g_1],\ldots,[g_n])=0_{\mathbb{G}_T}$ if $[g_i]\in\mathbb{H}^{(s)}$ and $[g_j]\in\mathbb{H}^{(s,\perp)}$ for any $i\neq j$.

Altogether, we have that $\tilde{e}:\mathbb{G}\times\cdots\times\mathbb{G}\to\mathbb{G}_T$ is a symmetric projecting and canceling k-linear map, where the group operations and $\tilde{e}$ can be efficiently computed. Note that the pairing now depends on s, hence our construction is *not a fixed pairing* as defined in Def. 6.We discuss the efficiency of our construction in Appendix C.

Choosing the polynomial $h'(X)$ In our construction, we made the choice of $h'(X)=X^n$ for reasons of efficiency. But our construction works for any fixed choice of $h'(X)$. In fact, we may even sample a random $h'(X)$ according to any distribution. If we do the latter, we may also keep $h'(X)$ secret along with s (which leads to a weaker security assumption). In any case, we may w.l.o.g. always assume that the constant coefficient of $h'(X)$ is 0, as this does not affect h.

[7]Note that doing it this way means computing exactly n^k basic pairings and is thus only efficient if n and k are constant. However, we stress that our construction becomes more efficient if we assume a "graded" k-linear map where we can compute intermediate results, i.e. products of less than k polynomials.

For our construction, if $h'(X)$ is secret, we still need to ensure that all $[h_{i,j}]$ are known, so we need to publish more hints (namely a subset of the $[h_{i,j}]$ from which all others can be computed) rather than only $[h'(s)], \ldots, [h'(s)^{k-1}]$, hurting efficiency even more. Our security proof in the generic group model supports any choice of $h'(X)$ (random or not, public or not), provided $h'(X)$ is sampled independently from s.

One issue that may appear is that for applications one might want $h(X)$ to split completely as $h(X) = (X-s_1)\cdots(X-s_n)$, as this affects the behaviour of orthogonality: In this case, one can have n non-zero vectors $[f_1], \ldots, [f_n]$ that are pairwise $\tilde{e}$-orthogonal by setting $f_i = \frac{h}{X-s_i}$. This means $\tilde{e}([f_i], [f_j], [g_3], \ldots, [g_{k-1}]) = [0]_T$ for all $i \neq j$ and arbitrary $g_3, \ldots, g_{k-1}$. For our choice of $h'(X) = X^n$, we have that $h(X) = (X-s)(X-\zeta s)\cdots(X-\zeta^{n-1}s)$ splits completely iff there exists a primitive nth root of unity $\zeta \in \mathbb{Z}_p$, i.e. $p \bmod n = 1$.

One might also directly sample $h(X) = (X-s_1)\cdots(X-s_n)$ for uniform choices of s_i. While not covered by our restriction that h be sampled as $h(X) = h'(X) - h'(s)$ with $(h'(X), s)$ independent, our security proof extends to that case, as discussed in Thm. 8 in Sec. B.4.

B.2 Implementation Using a k-linear Prime-Order Map

For our previous construction of a projecting and canceling k-linear map with a non-fixed pairing, a basic $k' = k+1$-linear prime-order map is required. We will now give a modification that only requires a k-linear prime order map. The tradeoff will be that our construction gives a $(k = k', (r, n, \ell = n-1))$ multilinear map generator as in Def. 5 with $r > n$, meaning that our group $\mathbb{G}$ rather than $\tilde{e}$ will depend on s and is embedded in some larger space $\mathbb{G} \subset G^r$, where $\tilde{e}$ is defined on G^r in a way independent from s.

Intuitively, one additional multiplication in the exponent is needed in order to perform the reduction, i.e., multiply products $f_{1,j_1} \ldots f_{k,j_k}$ of coefficients of $f_1, \ldots, f_k$ with coefficients $h_{i,j}$ for the reduction. If for one factor, say a_{1,j_1}, we were given $[a_{1,j_1} h_{i,j}]$ rather than $[a_{1,j_1}]$, this problem would not occur. To put us in this situation, we may consider first a simple extended version of $\mathbb{G}$: Let $\overline{\mathbb{G}_{\text{ext}}} \subset G^r$, where $r = (\kappa+1)\cdot n = kn$ and $\kappa = k-1$ is the number of hints we needed in the preceding construction, be defined as [8]

$$\overline{\mathbb{G}_{\text{ext}}} = \{([f], [s^n f], [s^{2n} f], \ldots, [s^{\kappa n} f]) \mid [f] \in \mathbb{G}\}$$

Similarly consider $\overline{\mathbb{H}_{\text{ext}}}^{(s)} \subset \overline{\mathbb{G}_{\text{ext}}}$, defined as $\overline{\mathbb{H}_{\text{ext}}}^{(s)} = \{[f], [fs^n], \ldots \in \overline{\mathbb{G}_{\text{ext}}} \mid f(s) = 0\}$. This just means that whoever initially computed $[f]$ in an application, computes and sends all $[f \cdot s^{n\cdot i}]$ alongside with it. We publish $[\mathbf{A}(s)], [s^n \mathbf{A}(s)], \ldots, [s^{\kappa n}\mathbf{A}(s)]$ to allow efficient sampling from $\mathbb{H}^{(s)}$. This contains $[s^n], [s^{2n}], \ldots, [s^\kappa n]$, allowing efficient sampling from $\mathbb{G}$. Testing membership in $\mathbb{G}$ is possible knowing $[s^n]$ using only a bilinear pairing. We still have $\dim \overline{\mathbb{G}_{\text{ext}}} = \dim \mathbb{G} = n$, but we redundantly use $r = (\kappa+1)n$ base group elements to represent elements from $\overline{\mathbb{G}_{\text{ext}}}$, i.e. this yields a $(k, (r = (\kappa+1)n, n, \ell = n-1))$ multilinear map generator as in Def. 5 with $r > n$. Our efficiently computable symmetric projecting and canceling k-linear map is

$$\tilde{e}_{\text{ext}} \colon \overline{\mathbb{G}_{\text{ext}}}^k \to \mathbb{G}_T \cong G^n,\ \tilde{e}_{\text{ext}}(([f_1], \ldots, [s^{\kappa n} f_1]), \ldots, ([f_k], \ldots, [s^{\kappa n} f_k])) = [f_1 \cdots f_k \bmod h]_T$$

or, more efficiently, a k-linear map $\overline{\mathbb{G}_{\text{ext}}} \times \mathbb{G}^{k-1} \to \mathbb{G}_T$.

In this construction, for every $[\vec{f}] \in \mathbb{G}$, we are provided with $[s^{in} f_j]$ for any $0 \leq i \leq \kappa, 0 \leq j < n$. But in fact, subsets of those are already sufficient to perform the multiplication and modular reduction. Restricting to such a subset can only improve security and reduces r, so we consider as our final proposal another extended version of $\mathbb{G}$, where we reduce r to $r = 2n-2$. Consider $\mathbb{G}_{\text{ext}} \subset G^{2n-2}$ and similarly $\mathbb{H}^{(s)}_{\text{ext}} \subset \mathbb{G}_{\text{ext}}$, defined as

$$\mathbb{G}_{\text{ext}} = \left\{ ([f], [s^n f_2], [s^n f_3], \ldots, [s^n f_{n-1}]) \;\middle|\; [f] \in \mathbb{G}, f = \sum_{i=0}^{n-1} f_i X^i \right\}$$

[8] for general public $h'(X)$, this is to be changed to $[f], [h'(s)f], \ldots [h'(s)^\kappa f]$ and for secret $h'(X)$ to a (subset of) $[f], [h_{0,0}f], \ldots, [h_{n(k-1),n-1}f]$, increasing κ.

To see that this works out, write the product $g = f_1 \cdots f_k$ of polynomials $f_i = \sum f_{i,j} X^j$ as $g = \sum_j g_j X^j$. To compute $g \bmod (X^n - s^n)$, we need to multiply each g_j by s^{in}, where $i = \lceil \frac{j}{n} \rceil$. Each g_j is a sum of terms $f_{1,j_1} \cdots f_{k,j_k}$ with $\sum_\ell j_\ell = j$, where one can easily verify that for $k < n$, in each such summand, we must have at least $\lceil \frac{j}{n} \rceil$ factors with $j_\ell \geq 2$. Consequently, we can compute $[g \bmod h]$ by picking up enough s^n-factors for each summand if we are given only $[s^n f_{i,j}]$ for $j \geq 2$. This yields an efficiently computable k-linear map

$$\tilde{e}_{\text{ext}} \colon {\mathbb{G}_{\text{ext}}}^k \to \mathbb{G}_T \cong G^n,\ \tilde{e}_{\text{ext}}(([f_1], \ldots, [s^n f_{1,n-1}]), \ldots, ([f_k], \ldots, [s^n f_{k,n-1}])) = [f_1 \cdots f_k \bmod h]_T$$

which is still both projecting and canceling. To allow sampling and membership testing we publish $[\mathbf{A}(s)]$ and $[s^n \mathbf{A}(s)]$. Given the concrete form of $\mathbf{A}(s)$, this means publishing $[s], [s^n], [s^{n+1}]$. Needing only a $k' = k$-linear basic map allows us to perform our construction based on (a modified version of) k-SCasc (rather than $k+1$-SCasc). The minimal interesting example with $k = 2, n = 3$ then reads as follows:

$$\tilde{e}_{\text{ext}}(([f_0], [f_1], [f_2], [s^3 f_2]), ([f_0'], [f_1'], [f_2'], [s^3 f_2'])) = \tag{10}$$

$$\Big([f_0 f_0' + f_1(s^3 f_2') + (s^3 f_2) f_1']_T, [f_0 f_1' + f_1 f_0' + (s^3 f_2) f_2']_T, [f_2 f_0' + f_1 f_1' + f_0 f_2']_T\Big) \tag{11}$$

This can be computed with only a 2-linear map using 9 basic pairing operations. In general, this construction requires n^k applications of e, both for $\mathbb{G}_{\text{ext}}$ and $\overline{\mathbb{G}_{\text{ext}}}$.

For $\overline{\mathbb{G}_{\text{ext}}}$ our security assumption changes into asking that

$$(\mathcal{MG}_k, [\mathbf{A}(s)], [s^n \mathbf{A}(s)], \ldots, [s^{\kappa n} \mathbf{A}(s)], [\mathbf{A}(s)\vec{w}], \ldots, [s^{\kappa n} \mathbf{A}(s)\vec{w}]) \quad \text{and}$$
$$(\mathcal{MG}_k, [\mathbf{A}(s)], [s^n \mathbf{A}(s)], \ldots, [s^{\kappa n} \mathbf{A}(s)], [\vec{u}], \ldots, [s^{\kappa n} \vec{u}])$$

be computationally indistinguishable for $\mathcal{MG}_k \leftarrow \mathcal{G}_k, s, \vec{w}, \vec{u}$ uniform. Note that the $[s^{in} \mathbf{A}(s)]$ given here are required to sample from $\overline{\mathbb{G}_{\text{ext}}}$ and $[s^n]$ is contained in $[s^n \mathbf{A}(s)]$, which allows to test membership in $\overline{\mathbb{G}_{\text{ext}}}$. The security assumption for $\mathbb{G}_{\text{ext}}$ is analogous and reads that

$$(\mathcal{MG}_k, [s], [s^n], [s^{n+1}], [\mathbf{A}(s)\vec{w}], [(s^n \mathbf{A}(s)\vec{w})_2], \ldots, [(s^n \mathbf{A}(s)\vec{w})_n]) \quad \text{and}$$
$$(\mathcal{MG}_k, [s], [s^n], [s^{n+1}], [\vec{u}], [s^n u_2], \ldots, [s^n u_n])$$

be computationally indistinguishable.

B.3 Proof of Generic Security of our Constructions

In this section, we show that our constructions from Sec. B.1 and Sec. B.2 are secure in a generic multilinear group model. Note that the following theorem also covers all our constructions where the distribution of $h'(X)$ might contain no randomness at all and the distinguisher $\mathcal{A}$ may only get a subset of the data $\{h'(X), [h'(s)], \ldots\}$ or something efficiently computable from that (like $[h_{i,j}]$), making it only harder for $\mathcal{A}$. In particular, setting $h'(X) := X^n$, it covers the Extended SCasc assumption from Def. 8.

Theorem 7. *Let $n > k$ and $\kappa \geq 0$ arbitrary but fixed. Let $\mathbf{A}(X)$ be the matrix associated to the $n-1$-SCasc assumption. Then for any algorithm $\mathcal{A}$ in the generic k-linear group model, making at most* $\text{poly}(\log p)$ *many oracle queries, its distinguishing advantage*

$$\eta = \big|\mathbf{Pr}[\mathcal{A}(p, h'(X), [h'(s)], \ldots, [h'(s)^\kappa], [\mathbf{A}(s)], [\mathbf{A}(s)\vec{w}], [h'(s)\mathbf{A}(s)\vec{w}], \ldots, [h'(s)^\kappa \mathbf{A}(s)\vec{w}]) = 1] -$$
$$\mathbf{Pr}[\mathcal{A}(p, h'(X), [h'(s)], \ldots, [h'(s)^\kappa], [\mathbf{A}(s)], [\vec{u}], \ldots, [h'(s)\vec{u}], \ldots, [h'(s)^\kappa \vec{u}) = 1]\big|$$

is negligible, where $h'(X) \leftarrow_\$ \mathbb{Z}_p[X]$ of degree n is sampled according to any distribution (but independent from $s, \vec{w}, \vec{u}$), $s \leftarrow \mathbb{Z}_p, \vec{w} \leftarrow \mathbb{Z}_p^{n-1}, \vec{u} \leftarrow \mathbb{Z}_p^n$ uniform.

Proof. W.l.o.g. we may assume $k = n - 1$. We may further assume that $h'(X)$ is a fixed, public polynomial, containing no randomness: Clearly, the distinguishing advantage η of $\mathcal{A}$ is the expected value (over the choice of $h'(X)$) of the conditional advantage $\mathbf{E}[\eta|h']$. Our argument will show that $\mathcal{A}$'s advantage for $h'(X)$ fixed is bounded by some negligible function, where the bound does not depend on $h'(X)$. This effectively means that we consider adversaries that may even depend non-uniformly on $h'(X)$.

Let us first consider the case where the distributions, which we want to show to be indistinguishable, are given by

$$(p, [h'(s)], [s], [\mathbf{A}(s)\vec{w}]) \text{ respectively}$$
$$(p, [h'(s)], [s], [\vec{u}])$$

The general case will follow as a by-product of our proof, as we will (almost) pretend that multiplying by $[h'(s)]$ is for free and does not consume a pairing, so $\mathcal{A}$ can compute the missing data itself. $h'(X)$ is a public constant and the entries of $[\mathbf{A}(s)]$ are either $[0], [1]$ (which we assume to be given as part of the group's description / oracle) or $[s]$.

Following [9], this implies that the assumption we are about to prove is polynomial-induced (i.e. the inputs to the adversary are obtained by evaluating bounded-degree polynomials in uniformly chosen unknowns). Consider the ideals

$$I_{\text{subgroup}} = \left(H - h'(S), S - X, \vec{Z} - A(S)\vec{W}\right) \in \mathbb{Z}_p[H, X, S, \vec{W}, \vec{Z}]$$
$$I_{\text{uniform}} = \left(H - h'(S), S - X\right) \in \mathbb{Z}_p[H, X, S, \vec{W}, \vec{Z}]$$

Here, $\vec{Z} - A(S)\vec{W}$ is shorthand for the n polynomial relations of a matrix-vector product, where $A(S)$ is the $n \times n - 1$ matrix of the SCasc assumption with polynomials as entries, so $A_{i,i} = -S, A_{i,i+1} = 1$ and $A_{i,j} = 0$ for $j \notin \{i, i + 1\}$.

The H-variable corresponds to the hint that make this different from the non-extended SCasc assumption, the X-variable corresponds to the known entries of the matrix A, the Z-variables to either $[A\vec{w}]$ or $[\vec{u}]$ and $\vec{W}, S$ are the uniformly chosen unknowns. $\vec{W}, S$ are only accessible to the adversary via the relations from the ideals. Note that these two ideals encode all relationships between these data.

Consider the ideals $J_{\text{subgroup}} = I_{\text{subgroup}} \cap \mathbb{Z}_p[H, X, \vec{Z}]$, $J_{\text{uniform}} = I_{\text{uniform}} \cap \mathbb{Z}_p[H, X, \vec{Z}]$, which encode the relations in those variables $(H, X, \vec{Z})$ that the adversary sees. By [9, Theorem 3] (which was only proven for matrix assumptions, but the statement and proof extend directly to our setup), it suffices to show that $J_{\text{subgroup},\leq n-1} = J_{\text{uniform},\leq n-1}$, the subscripts denoting restriction to total degree $\leq n - 1$.

As mentioned briefly above we will strengthen the adversary and allow it to compute polynomials $p(H, \vec{Z}, X)$ of degree totaling at most $k = n - 1$ in $(\vec{Z}, X)$, i.e. we lift any degree restrictions on H. This means showing $J_{\text{subgroup},(\vec{Z},X)\text{-degree } \leq n-1} = J_{\text{uniform},(\vec{Z},X)\text{-degree } \leq n-1}$.

Translated back from the language of ideals to generic algorithms, this corresponds to allowing the adversary to multiply by $h'(S)$ for free (provided the degrees of all polynomials appearing remain bounded), thereby allowing it to compute the missing data. As a side remark, the bound κ (which gives a restriction on how $\mathcal{A}$ is allowed to multiply by $h'(S)$) is required, because otherwise equality of those ideals, restricted by degrees, no longer is equivalent to generic security (and hence the translation back from the language of ideals to generic algorithms fails). Working through [9, Theorem 3] gives a bound on the distinguishing advantage via the Schwarz-Zippel lemma, which depends on the maximal degree of any polynomial that can appear. To ensure this bound is negligible, we need κ to be constant (and the bound is uniform in $h'(X)$). Still, we can forget about κ in our proof here from now on.

To compute J_{subgroup} and J_{uniform} from I_{subgroup} and I_{uniform}, we need to eliminate the S and $\vec{W}$-variables. Elimination of S means just using $X - S$ to plug in X for S. Elimination of the $\vec{W}$-variables can be done as in the security proof of the non-extended SCasc (the additional hint H does not affect that part of the

proof), so we have

$$J_{\text{subgroup}} = \Big(H - h'(X), \mathfrak{d}(Z, X)\Big)$$
$$J_{\text{uniform}} = \Big(H - h'(X)\Big)$$

where $\mathfrak{d}(Z, X) = \pm Z_0 \pm Z_1 X \pm \ldots \pm Z_{n-1} X^{n-1}$ is the determinant polynomial of SCasc for some specific choice of signs. It was shown in [9] to be absolutely irreducible.

Of course, $J_{\text{uniform}} \subset J_{\text{subgroup}}$. So, assume towards a contradiction that there exists some adversarially computable polynomial $\mathfrak{p}(H, \vec{Z}, X) \in J_{\text{subgroup}} \setminus J_{\text{uniform}}$ of total degree in $\vec{Z}, X$ at most $k = n - 1$. By definition this implies that there exist polynomials $\mathfrak{a}, \mathfrak{b} \in \mathbb{Z}_p[H, \vec{Z}, X]$ such that

$$\mathfrak{p}(H, \vec{Z}, X) = \mathfrak{a}(H, \vec{Z}, X) \cdot \mathfrak{d}(\vec{Z}, X) + \mathfrak{b}(H, \vec{Z}, X) \cdot (H - h'(X)) \tag{12}$$

The existence of $\mathfrak{b}$ in the above equation (for $\mathfrak{p}, \mathfrak{a}$ fixed) is equivalent to just plugging in $h'(X)$ for any occurrence of H, so we have

$$\mathfrak{p}'(\vec{Z}, X) := \mathfrak{p}(h'(X), \vec{Z}, X) = \mathfrak{a}(h'(X), \vec{Z}, X) \cdot \mathfrak{d}(\vec{Z}, X) = \mathfrak{a}'(\vec{Z}, X) \cdot \mathfrak{d}(\vec{Z}, X) \tag{13}$$

where $\mathfrak{a}'(\vec{Z}, X) := \mathfrak{a}(h'(X), \vec{Z}, X) \neq 0 \in \mathbb{Z}_p[\vec{Z}, X]$, as otherwise $\mathfrak{p} \in J_{\text{uniform}}$.

Let us give some intuition what we need to show here. The theorem from [9] essentially says that the only thing the adversary can do if the determinant $\mathfrak{d}$ is irreducible is to compute this determinant or a multiple thereof. This remains true in our case. For the usual SCasc assumption this was easily shown to be impossible, because the determinant had a higher degree than anything the adversary could compute. In our case, the situation changes, because the adversary has the polynomial H, which corresponds to S^n, at its disposal and S^n has the same degree as $\mathfrak{d}$. It is actually still easy to show that the adversary can not compute $\mathfrak{d}$ itself. The real problem is to show that this also holds for multiples $\mathfrak{a}(h'(X), \vec{Z}, X) \cdot \mathfrak{d}(\vec{Z}, X)$.

To prove our theorem, we will show that indeed $\mathfrak{a}' = 0$ is the only solution of Eq. (13), even when extending the base field to an algebraic closure $\overline{\mathbb{Z}_p}$.
Let us make another assumption simplifying the proof: Changing $h'(X)$ into $h'(X) + c$ for any constant $c \in \overline{\mathbb{Z}_p}$ does not affect the statement of the theorem. By using such a change, we may assume that $h'(X)$ is square-free. Note here that the condition that $h'(X) + c$ be square-free is equivalent to requiring the discriminant of $h'(X) + c \neq 0$. The discriminant of $h'(X) + c$ is a polynomial in c, which equals the resultant $\mathrm{Res}_X(\frac{\mathrm{d}h'}{\mathrm{d}X}, h'(X) + c)$ up to some normalization constant. Computing the determinant of the Sylvester matrix for this resultant results in a leading term of $n^n c^{n-1}$, so the discriminant does not vanish identically and we can find a value c (in the base field, even, if $n \geq p$) such that $h'(X) + c$ is square-free.

Let $a_0, \ldots, a_{n-1}$ be the n distinct roots of $h'(X)$ in $\overline{\mathbb{Z}_p}$.

After performing a linear, invertible change of variables (which does not affect anything at hand here), we may consider the variables $\vec{Z}'$ instead of $\vec{Z}$, defined by $Z'_i = \mathfrak{d}(\vec{Z}, a_i) = \sum_j \pm Z_j \cdot a_i^j$ and express everything in terms of $\vec{Z}'$, redefining $\mathfrak{p}, \mathfrak{p}', \mathfrak{a}, \mathfrak{a}', \mathfrak{d}$ accordingly as if we had made h' square-free and expressed everything in terms of $\vec{Z}'$ from the beginning. This will simplify things later, as now $\mathfrak{d}(\vec{Z}', a_i) = Z'_i$. Note here that the matrix of the linear map relating $\vec{Z}'$ and $\vec{Z}$ is a Vandermonde matrix and hence invertible.

Our proof will proceed in two steps. In the first step, we will show that if h' divides $\mathfrak{p}'$ (which corresponds to $\mathfrak{p}(H, \vec{Z}, X)$ being a multiple of H), then we can divide everything by H to obtain another non-trivial solution of 13 with smaller degrees. In the second step, we will show that it is always the case the h' divides $\mathfrak{p}'$. This leads to a contradiction.

For the first step, let us consider the case where h' divides $\mathfrak{p}'$ and consequently h' divides $\mathfrak{a}' \cdot \mathfrak{d}$. Since $\overline{\mathbb{Z}_p}[H, \vec{Z}', X]$ is factorial, $\mathfrak{d}(\vec{Z}', X)$ is absolutely irreducible and h' can't divide $\mathfrak{d}$ for degree reasons, this means that h' must divide $\mathfrak{a}'$. In this case, we may divide both $\mathfrak{p}'$ and $\mathfrak{a}'$ by h' to obtain another solution $(\widetilde{\mathfrak{p}}', \widetilde{\mathfrak{a}}')$ of (13) with $\mathfrak{a}' = \widetilde{\mathfrak{a}}' \cdot h', \mathfrak{p}' = \widetilde{\mathfrak{p}}' \cdot h'$. Note that we can uniquely recover $\mathfrak{p}$ from $\mathfrak{p}'$ due to the degree restriction: For any polynomial $\mathfrak{f} \in \overline{\mathbb{Z}_p}[\vec{Z}', X]$, let $C(\mathfrak{f}) \in \overline{\mathbb{Z}_p}[H, \vec{Z}', X]$ be the *unique* polynomial of degree

at most $n-1$ in X, such that $C(\mathfrak{f})(h'(X), \vec{Z}', X) = \mathfrak{f}(\vec{Z}', X)$. By uniqueness, we have $H \cdot C(\widetilde{\mathfrak{p}}') = C(h' \cdot \widetilde{\mathfrak{p}}')$. Consequently, $\mathfrak{p} = C(\mathfrak{p}') = C(h' \cdot \widetilde{\mathfrak{p}}') = H \cdot C(\widetilde{\mathfrak{p}}')$. This means that H divides $\mathfrak{p}$ and we may divide $\mathfrak{p}$ by H to obtain $\widetilde{\mathfrak{p}}$, such that $(\widetilde{\mathfrak{p}}, C(\widetilde{\mathfrak{a}}'))$ is another solution of (12). Note that by construction $\widetilde{\mathfrak{p}}$ still satisfies the degree restrictions and $\widetilde{\mathfrak{a}}' \neq 0$. After performing this transformation from $\mathfrak{p}$ to $\widetilde{\mathfrak{p}}$ finitely many times, we are in the case where h' does not divide $\mathfrak{p}'$, so we may w.l.o.g. assume from now on that h' does not divide $\mathfrak{p}'$.

We now show in the second step that h' divides $\mathfrak{p}'$, leading to a contradiction. For this, we take Equation (13) modulo h'

$$\mathfrak{p}''(\vec{Z}', X) := \mathfrak{p}'(\vec{Z}', X) \bmod h' = \mathfrak{p}(0, \vec{Z}', X) = (\mathfrak{a}'(\vec{Z}', X) \cdot \mathfrak{d}(\vec{Z}', X)) \bmod h'$$

The degree restrictions on $\mathfrak{p}$ imply that $\mathfrak{p}''$ is now a polynomial of total degree at most $n-1$.

Let us plug in a_i for X in both sides of this equation. Since a_i was defined as a root of h' we have $(\mathfrak{f} \bmod h')(a_i) = \mathfrak{f}(a_i)$ and by definition of the Z_i''s in terms of Z_i, we obtain:

$$\mathfrak{p}''(\vec{Z}', a_i) = \mathfrak{a}'(\vec{Z}', a_i) \cdot \mathfrak{d}(\vec{Z}', a_i) = \mathfrak{a}'(\vec{Z}', a_i) \cdot Z_i', \qquad \text{for all } 0 \leq i \leq n-1 \tag{14}$$

Now consider the coefficient $c_{\vec{\alpha}} \in \overline{\mathbb{Z}_p}[X]$ of $\vec{Z}'^{\vec{\alpha}}$ in $\mathfrak{p}''(\vec{Z}', X)$. Since $\mathfrak{p}''(\vec{Z}', a_i)$ is divisible by Z_i', we must have $c_{\vec{\alpha}}(a_i) = 0$ whenever $\alpha_i = 0$. If $|\alpha|_1 = \gamma$, there are at least $n - \gamma$ indices i, such that $\alpha_i = 0$. Consequently, $c_{\vec{\alpha}}$ has at least $n - \gamma$ distinct roots. But our degree restriction on $\mathfrak{p}''$ means that $c_{\vec{\alpha}}$ can have degree at most $n - 1 - \gamma$. Hence all $c_{\vec{\alpha}}$ are 0 and $\mathfrak{p}'' = 0$. This in turn means that h' divides $\mathfrak{p}'$, which we ruled out above, giving us a contradiction. This shows that such a $\mathfrak{p}$ can't exist, finally finishing the proof. □

B.4 Generic Security For h Composed of Random Linear Factors

In Sec. B.1, we discussed that it might be desirable to choose h as $h(X) = (X - s_1) \cdots (X - s_n)$, where s_1 corresponds to s. This is not of the form $h(X) = h'(X) - h'(s_1)$ where $h'(X)$ is sampled independently from $s = s_1$ and hence our proof above does not directly apply to this case. However, the case $h(X) = (X - s_1) \cdots (X - s_n)$ is essentially equivalent to setting $h'(X)$ uniform, conditioned on the event that $h'(X) - h'(s_1)$ splits completely over the base field. Intuitively, we expect that conditioning on the event that $h'(X) - h'(s_1)$ splits completely can not change generic security. The reason is that generic security can be expressed as an equality of ideals up to some degree as in [9] or by the Uber-Assumption Theorem from [1, 5]. In any case, it boils down to a problem of linear algebra, which does not depend on whether we are in $\mathbb{Z}_p$ or in the algebraic closure $\overline{\mathbb{Z}_p}$ and in the latter case, every polynomial splits completely. Rather than making this precise, we will show the stronger statement that security of choosing $h = (X - s_1) \cdots (X - s_n)$ is implied by security of choosing $h = h'(X) - h'(s_1)$, h' uniform *in the standard model*.

Theorem 8. *Let $n > k$ and let $\mathcal{G}_k$ be a symmetric prime-order k-linear group generator. Consider a PPT adversary $\mathcal{A}$ with advantage*

$$\eta = \big|\mathbf{Pr}\big[\mathcal{A}(\mathcal{MG}_k, [h_{0,0}], \ldots, [h_{k(n-1),n-1}], [\mathbf{A}], [\mathbf{A}\vec{w}]) = 1\big] - \tag{15}$$
$$\mathbf{Pr}\big[\mathcal{A}(\mathcal{MG}_k, [h_{0,0}], \ldots, [h_{k(n-1),n-1}], [\mathbf{A}], [\vec{u}]) = 1\big]\big| \tag{16}$$

where $\mathcal{MG}_k := (k, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_k(1^\lambda)$, $s_1, \ldots, s_n \in \mathbb{Z}_p, \vec{u} \in \mathbb{Z}_p^n, \vec{w} \in \mathbb{Z}_p^{n-1}$ uniform, $h(X) = (X - s_1) \cdots (X - s_n)$ and $h_{i,j}$ is the jth coefficient of $X^i \bmod h$. Assume $\eta > \frac{1}{\mathrm{poly}(\lambda)}$.

Then there exists another PPT adversary $\mathcal{A}'$ with

$$\eta' = \big|\mathbf{Pr}\big[\mathcal{A}'(\mathcal{MG}_k, [h_{0,0}], \ldots, [h_{k(n-1),n-1}], [\mathbf{A}], [\mathbf{A}\vec{w}]) = 1\big] -$$
$$\mathbf{Pr}\big[\mathcal{A}'(\mathcal{MG}_k, [h_{0,0}], \ldots, [h_{k(n-1),n-1}], [\mathbf{A}], [\vec{u}]) = 1\big]\big| > \mathrm{negl}(\lambda)$$

where $\mathcal{MG}_k := (k, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T) \leftarrow \mathcal{G}_k(1^\lambda)$, $s_1 \in \mathbb{Z}_p, \vec{u} \in \mathbb{Z}_p^n, \vec{w} \in \mathbb{Z}_p^{n-1}$ uniform, $h'(X) \leftarrow \mathbb{Z}_p[X]$ is a uniformly chosen polynomial of degree n with leading coefficient 1 and constant coefficient 0, $h(X) = h'(X) - h'(s_1)$ and $h_{i,j}$ is the jth coefficient of $X^i \bmod h$.

Proof. With overwhelming probability, the s_i in the first variant are pairwise different and in the second variant $h = h'(X) - h'(s_1)$ is square-free. So it is sufficient to consider the (statistically close) variants, where we sample $(s_1, \ldots, s_n)$ uniform, conditioned on being pairwise different, respectively (h', s_1) uniform, conditioned on $h'(X) - h'(s_1)$ square-free. For $s_1, \ldots, s_n$ pairwise different, the map

$$(s_1, \ldots, s_n) \mapsto (s_1, h' = (X - s_1) \cdots (X - s_n) - s_1 \cdots s_n) \tag{17}$$

is exactly $(n-1)!$ to 1. As a consequence, if we sample $(h'(X), s_1)$, conditioned on the event $\texttt{Split}$ that $h'(X) - h'(s_1)$ completely splits over $\mathbb{Z}_p$, we obtain exactly the same distribution on h as if we had sampled h as $h(X) = (X - s_1) \cdots (X - s_n)$. Further, we have $\mathbf{Pr}[\texttt{Split}] = \frac{1}{(n-1)!}$. By a standard argument, there exists at least an $\frac{\eta}{2}$-fraction of "good" choices of h in the first variant, where the advantage of $\mathcal{A}$, conditioned on this h, is at least $\frac{\eta}{2}$.

As a consequence, simply running $\mathcal{A}$ on the second variant will give us a conditional advantage of at least $\frac{\eta}{2}$ for at least a $\frac{\eta}{2 \cdot (n-1)!}$ - fraction of "good" choices of h. For other values of h, we can simply guess to obtain an advantage of 0. Unfortunately, we cannot easily detect whether we have a good h. However, we can define $\mathcal{A}'$ as follows: First run a statistical test, which outputs 1 with overwhelming probability if the conditional advantage of $\mathcal{A}$ for the given h is at least $\frac{\eta}{4}$ and outputs 0 with overwhelming probability, if the conditional advantage of $\mathcal{A}$ is at most $\frac{\eta}{8}$. If this test outputs 1, $\mathcal{A}'$ can simply use $\mathcal{A}$ to output its final answer, otherwise $\mathcal{A}'$ just guesses. Note that since $\eta > \frac{1}{\text{poly}(\lambda)}$, such a statistical test can be performed in probabilistic polynomial time, using the fact that $\mathcal{A}'$ can create instances for given $\mathcal{MG}_k, [h_{i,j}], [\mathbf{A}]$ itself by sampling its own $\vec{w}$'s respectively $\vec{u}$'s. Also, note that this reduction is not black-box, because the code of $\mathcal{A}'$ depends on the advantage η. □

C Efficiency Considerations for our Constructions

In some instantiations of multilinear settings, computing the (basic) mapping e is significantly more expensive than computing the group operation or even an exponentiation. For instance, this is the case for all instantiations of bilinear maps over elliptic curves we currently know. In such settings it might be worthwhile to strive for tradeoffs between applications of e and less expensive operations. In Sec. C.1 we consider such tradeoffs for our projecting map constructions while Sec. C.2 deals with the canceling & projecting maps.

C.1 Efficiency of the Projecting Constructions

For our constructions of projecting maps in Sec. 4.1 and Sec. 4.2, computing $\tilde{e}$ corresponds to usual multiplication of polynomials. Hence, we may apply methods for fast polynomial multiplication to reduce the number of applications of e. Concretely, we may follow an evaluate-multiply-interpolate approach. Consider the case that we are given the polynomials $f_1, \ldots, f_k$ all from a subspace V of dimension n (e.g., univariate polynomials of degree at most $n-1$), and we know that their product lies in a subspace W of dimension m (e.g., $k = 2$ and W contains all univariate polynomials of degree at most $2n-2$, so $m = 2n-1$). Then we can first evaluate all f_j at m publicly known points $x_0, \ldots, x_{m-1}$ that form an interpolating set for W, then multiply $f_1(x_i) \cdots f_k(x_i)$ for any i to obtain $(f_1 \cdots f_k)(x_i)$, from which our desired result $f_1 \cdots f_k$ can be interpolated. One thing to note here is that the map sending a polynomial $f \in W$ to the vector $f(x_0), \ldots, f(x_{m-1})$ is a bijective linear map, whose coefficients depend only on the publicly known x_i, so both evaluation and interpolation can be computed without any pairings. As a consequence, using this approach for computing $\tilde{e}$, we can reduce the number applications of e to m at the cost of having to apply some linear maps, which correspond to multi-exponentiations in G.

Intermediate tradeoffs are possible here. For instance, it is easy to see that the Karatsuba algorithm [16] can immediately be applied to our bilinear map based on 2-SCasc. This would reduce the number of basic pairing applications from a naive 9 to 6 (rather than all the way to 5) at the cost of only 9 additional group operations (e.g., see [26]). Note that there are also generalizations of Karatsuba to the multivariate case, e.g., [27].

Since interpolation is a publicly known linear map, this just corresponds to choosing a basis for W and has no effect on the hardness of subgroup indistinguishability. In particular, this means we do not need to interpolate in the end, but can simply use $[(f_1 \cdots f_k)(x_0)]_T, \ldots, [(f_1 \cdots f_k)(x_{m-1})]_T \in G_T^m$ as the final result, representing polynomials in the target space by their evaluations at the interpolating set.

This observation means that we should choose the interpolation points in such a way that computations of the map $\mathbb{G} \to G^m, [f] \mapsto ([f(x_0)], \ldots, [f(x_{m-1}])$ should be cheap. If vectors from $\mathbb{G}$ correspond to polynomials via coefficient representation, we can simply choose small x_i (usually, this has the downside of making the coefficients for interpolation large, which does not matter here). Concretely, for our 2-SCasc based construction, we chose interpolation points as $M = \{-2, -1, 0, 1, 2\}$. Given coefficients $[f_0], [f_1], [f_2]$ of a polynomial $f = f_0 + f_1 X + f_2 X^2$ of degree at most 2, one can compute $[f(-2)], [f(-1)], [f(0)], [f(1)], [f(2)]$ with only 11 additions/inversions. Furthermore, it is possible to amortize the cost of evaluation, if the same $[f]$ is used in several applications of $\tilde{e}$.

Computing $\pi\colon \mathbb{G} \to G$ for known $\vec{s}$ is a linear map and hence corresponds to one n-multi-exponentiation. For our SCasc-based constructions, this means computing $[f_0 + sf_1 + s^2 f_2 + \ldots]$ from $[f]$ and s. Since f_0 is not multiplied by anything, we really only have a $n-1$-multi-exponentiation and one group operation. For the computation of $\pi_T\colon \mathbb{G}_T \to G_T$, this latter saving is no longer possible if we represent elements from W by their evaluations. Instead, $[f(s)]_T = \pi_T([g_0]_T, \ldots, [g_{m-1}]_T)$ is computed as $\pi_T([g_0]_T, \ldots, [g_{m-1}]_T) = \sum_i \mathfrak{r}_i(\vec{s}) \cdot [g_i]_T$, where the coordinate $[g_i]_T$ corresponds to the value $g_i = f(x_i)$ of some polynomial f at x_i. This corresponds to the basis $\mathfrak{r}_0, \ldots, \mathfrak{r}_{m-1}$ of W, determined by $\mathfrak{r}_i(x_j) = 0$ if $i \neq j$ and 1 otherwise. Note that the $\mathfrak{r}_i$ are known and computing $\mathfrak{r}_i(\vec{s})$ is just a computation in $\mathbb{Z}_p$ (which is fast).

C.2 Efficiency of the Projecting and Canceling Constructions

Let us briefly consider our projecting and canceling constructions from Sec. B.1 and Sec. B.2 based on variants of SCasc. Computation of π and π_T can be done as in the projecting construction. So let us turn our attention to the efficiency of $\tilde{e}$ respectively $\tilde{e}_{\text{ext}}$ with respect to the application of fast multiplication algorithms. Here the the situation is more intricate as we also need to perform modular reduction in the exponent. Furthermore, we chose $h'(X) = X^n$, which gives us an advantage if we stay in the coefficient representation, as the reduction modulo $X^n - s^n$ has an easier form then.

The naive way of computing either $\tilde{e}$ or $\tilde{e}_{\text{ext}}$ requires exactly n^k applications of the k'-linear e and $n^k - n^{k-1}$ additions in G_T. For $\tilde{e}_{\text{ext}}$ from Sec. B.2, this is the best method we are aware of, both in the $\overline{\mathbb{G}_{\text{ext}}}$ and in the $\mathbb{G}_{\text{ext}}$ variant.

For $\tilde{e}$ from Sec. B.1, we can use some ideas from efficient polynomial multiplication to improve this. Perhaps, the most simple idea which, however, only works in certain settings is the following: Let us first assume that we are given a k'-linear basic map e to implement our k-linear map $\tilde{e}$ as in Sec. B.1. Moreover, assume that e is not given as a "monolithic block" but as a series of pairings $e_{i,j} : G^{(i)} \times G^{(j)} \to G^{(i+j)}$ like it is the case for the currently known multilinear map candidates. In such a setting, it is possible to first compute products consisting of only k factors and then multiply (linear combinations of) these subproducts with another factor. This enables us to first compute the coefficients of $[(f_1 \cdots f_k)]$ in $G^{(k)}$ using the fast polynomial multiplication algorithms as described before and subsequently, perform the modular reduction by multiplying these coefficients with the appropriate reduction term $[s^{in}]$ for appropriate i by means of $e_{k,1}$. Note that we can perform polynomial interpolation onto intermediate results, which means we can use a multiplication tree, reducing the number of interpolation points required for intermediate products. Also, we interpolate in the end, so the final modular reduction can be performed in the coefficient representation. This way, (only counting applications of e), for the multiplication, we need at most (or exactly if k is a power of 2) $k(n-1)\lceil \log_2 k \rceil + k - 1$ applications of some $e_{i,j}$, and for the reduction we need $k(n-1) - n$ applications of $e_{k,1}$. This makes a total of (at most) $k(n-1)\lceil 1 + \log_2 k \rceil + k - n - 1$ (bilinear) pairings. Note that this counts bilinear pairings, i.e. only "partial" k'-linear pairings and hence can not be directly compared to applications of a k'-linear map.

Now, assume we are given a $k+1$-linear basic map as a black-box, i.e., not as a series of pairings. We use the evaluate-multiply approach as before, so consider the interpolating set $x_0, \ldots, x_{k(n-1)}$ with interpolation

polynomials $\mathfrak{r}_i$ such that $\mathfrak{r}_i(x_j) = 0$ for $i \neq j$ and 1 otherwise. Let

$$\mathfrak{r}_i \bmod h = \sum_{j=0}^{n-1} \overline{h}_{i,j} X^j \ ,$$

Note that the $[\overline{h}_{i,j}]$'s are computable from the $[h_{i,j}]$'s and x_i's. Then we can compute $\tilde{e}([f_1], \ldots, [f_k])$ as

$$\tilde{e}([f_1], \ldots, [f_k]) = [f_1 \cdots f_k \bmod h]_T = \left[\sum_{j=0}^{n-1} \sum_{i=0}^{k(n-1)} \overline{h}_{i,j} f_1(x_i) \cdots f_k(x_i) X^j \right]_T \ .$$

This requires $kn(n-1) + n$ applications of e. Note that we can not make use of the special form of $h(X) = X^n - s^n$ this way and this is worse than the naive approach for small values of n, k (but much better asymptotically). Also for small values of n and k and h of a general form, there are dedicated tricks to reduce the number of basic map applications. For instance, in the case $k = 2$, $n = 3$, and general h, we may compute $\tilde{e}_{\text{ext}}$ (which is defined in a similar way as our construction in Sec. B.2 for special $h = X^n - s^n$) using 12 applications of e compared to 15 using the naive approach.

D Extended Impossibility Results for Canceling and Projecting

Freeman *et al.* [20] proved that there is no fixed projecting and canceling pairing for the $\mathcal{U}_\ell$ assumption. It could be the case that, as it happened for the lower bounds for the image size, a change of assumption could suffice to construct a projecting and canceling pairing. However, the proof of [20] seems hard to generalize to other $\mathcal{D}_{n=\ell+1,\ell}$ assumptions. In this section, we give a very simple but limited extension of Freeman's result. We start by proving the following lemma:

Lemma 9. *Let $(k = 2, \mathbb{H}_1, G^n, \mathbb{G}_T, \tilde{e})$ be the output of a symmetric canceling $(k = 2, (n = \ell + 1, \ell))$ bilinear map generator. Then $\tilde{e}(\mathbb{G}^k) \subset \mathbb{G}_T$ is a vector space of dimension at most $\ell(\ell+1)/2$.*

Proof. The map $\tilde{e}$ can be alternatively defined as a linear map from $\mathbb{G} \otimes \mathbb{G} \to \mathbb{G}_T$. First we note that, since $\tilde{e}$ is symmetric, the maximum dimension of the image of $\tilde{e}$ (which w.l.o.g. is G_T^m, for some $m \in \mathbb{N}$) is $(\ell+1)(\ell+2)/2$. This follows because the kernel of $\tilde{e}$ must contain all the symmetry relations , i.e. the span of all $\vec{e}_i \otimes \vec{e}_j - \vec{e}_j \otimes \vec{e}_i$. Additionally, since the map is canceling, and $\mathbb{G} = \mathbb{H}_1 \oplus \mathbb{H}_2$, it follows that $\mathbb{H}_1 \otimes \mathbb{H}_2$ must also be in the kernel (note that if this is the case, by symmetry so is $\mathbb{H}_2 \otimes \mathbb{H}_1$). Since $\mathbb{H}_1 \cap \mathbb{H}_2 = \{0\}$, we have that $\mathbb{H}_1 \otimes \mathbb{H}_2$ intersects the span of the symmetry relations only trivially. Since the dimension of $\mathbb{H}_1 \otimes \mathbb{H}_2$ is ℓ, it follows that the size of the image is at most $m := \frac{(\ell+1)(\ell+2)}{2} - \ell = \frac{\ell(\ell+1)}{2} + 1$. □

The lemma also means that there is no $(2, \mathcal{L}_\ell)$ bilinear generator with a fixed pairing which is both canceling and projecting, because according to Sec. 5.1 the image size would be at least $\frac{(\ell+1)(\ell+2)}{2}$, while Thm. 9 says the image is at most $\frac{\ell(\ell+1)}{2} + 1$.[9] Further, we can prove that there is no $(2, \mathcal{SC}_2)$ bilinear generator with a fixed pairing which is both canceling and projecting (more generally, this extends to any $\mathcal{D}_{3,2}$ matrix distribution), since the optimality results of Sec. 5.1 and 5.2 imply that the image size would be at least 5 while Thm. 9 says the image size would be at most 4. It remains an open question to see if other impossibility results for ℓ-SCasc can be proven for $\ell > 2$.

E Proofs of Optimality

E.1 Optimality of Polynomial Multiplication

We give the complete proof of Thm. 2, which states:
Let $k > 0$, $\mathcal{D}_{\ell+1,\ell}$ a polynomially induced matrix assumption and let $\mathfrak{q}_0, \ldots, \mathfrak{q}_\ell$ be the polynomials associated

[9] We note that this last result about $(2, \mathcal{L}_\ell)$ bilinear generators is not proven in [20]. Although the authors talk about a natural use of the ℓ-Lin assumption, their results are for the uniform assumption.

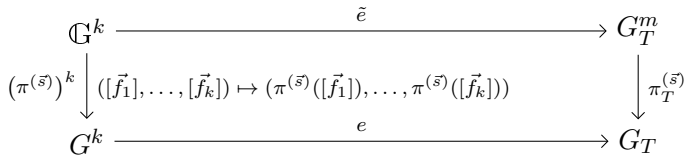

Figure 2: Projecting Property

to $\mathcal{D}_{\ell+1,\ell}$ as defined in Eq. (4) in Sec. 4.2 and let $W \subset \mathbb{Z}_p[\vec{X}]$ be the space of polynomials spanned by $\{\mathfrak{q}_{i_1} \ldots \mathfrak{q}_{i_k} \mid 0 \leq i_j \leq \ell\}$. Let $(\mathcal{MG}_k, \mathbb{H}, G^{\ell+1}, G_T^m, \tilde{e})$ be the output of any other fixed $(k, \mathcal{D}_{\ell+1,\ell})$ projecting multilinear map generator. Then, $\overline{m} := \dim W \leq m$.

By assumption, $\mathbb{H} = \mathbb{H}^{(\vec{s})}$ is the subspace of $\mathbb{G} = G^{\ell+1}$ spanned by the rows of the matrix $[\mathbf{A}(\vec{s})]$, for some $\vec{s} \in \mathbb{Z}_p^d$, and by definition of $\mathfrak{q}_0, \ldots, \mathfrak{q}_\ell$, if $[\mathbf{A}(\vec{s})]$ has full rank, $\mathbb{H}^{(\vec{s})} = \{\vec{f} = (f_0, \ldots, f_\ell) \mid \sum_{i=0}^{\ell} f_i \mathfrak{q}_i(\vec{s}) = 0\}$.

The fact that the map is projecting (cf. Def. 4 or Fig. 2) guarantees that for every $\mathbb{H}^{(\vec{s})}$ there exist $\pi^{(\vec{s})} \colon \mathbb{G} \to G$, and $\pi_T^{(\vec{s})} \colon \mathbb{G}_T \to G_T$, such that $\ker \pi^{(\vec{s})} = \mathbb{H}^{(\vec{s})}$ and $e(\pi^{(\vec{s})}(\vec{x}_1), \ldots, \pi^{(\vec{s})}(\vec{x}_k)) = \pi_T^{(\vec{s})}(\tilde{e}(\vec{x}_1, \ldots, \vec{x}_k))$ for any $\vec{x}_1, \ldots, \vec{x}_k$, where e is the basic pairing operation in $\mathcal{MG}_k$. We stress that $\pi^{(\vec{s})}$ and $\pi_T^{(\vec{s})}$ may depend on $\mathbb{H}$, while by assumption, the multilinear map $\tilde{e}$ is fixed and thus independent of $\mathbb{H}$.

We structure the proof into two steps: The first step is a lemma, which says that $\pi^{(\vec{s})}$ can be viewed as polynomial evaluation at $\vec{s}$.

Lemma 10. *For any $\vec{s} \in \mathbb{Z}_p^d$, there exists some $\mu^{(\vec{s})} \in \mathbb{Z}_p^*$ such that $\pi^{(\vec{s})}(\vec{f}) = \mu^{(\vec{s})} \sum_{i=0}^{\ell} f_i \mathfrak{q}_i(\vec{s})$.*

Proof. Since $\mathbb{H}^{(\vec{s})}$ has co-dimension 1 in $G^{\ell+1}$, any two maps $G^{\ell+1} \to G$, both with kernels $\mathbb{H}^{(\vec{s})}$, differ by a non-zero scalar multiple. By definition, $\ker \pi^{(\vec{s})} = \mathbb{H}^{(\vec{s})}$. Since the map $\tilde{\pi} \colon G^{\ell+1} \to G$ which sends $\vec{f} \in G^{\ell+1}$ to $f(\vec{s}) = \sum_{i=0}^{\ell} f_i \mathfrak{q}_i(\vec{s})$ is another linear map with kernel $\mathbb{H}^{(\vec{s})} = \{\vec{f} \in G^{\ell+1} \mid f(\vec{s}) = 0\}$, the claim follows. □

Without loss of generality we assume in the following that $\mu^{(\vec{s})} = 1$ for all $\vec{s}$. This follows from the fact that if $\tilde{e}$ satisfies the projecting property with respect to the maps $\pi^{(\vec{s})}, \pi_T^{(\vec{s})}$ then the same property is satisfied by the maps $((\mu^{(\vec{s})})^{-1}\pi^{(\vec{s})}), ((\mu^{(\vec{s})})^{-k}\pi_T^{(\vec{s})})$.
For the second step, we consider the commutative diagram Fig. 3 for an interpolating set $\vec{s}_1, \ldots, \vec{s}_{\overline{m}}$ for W:

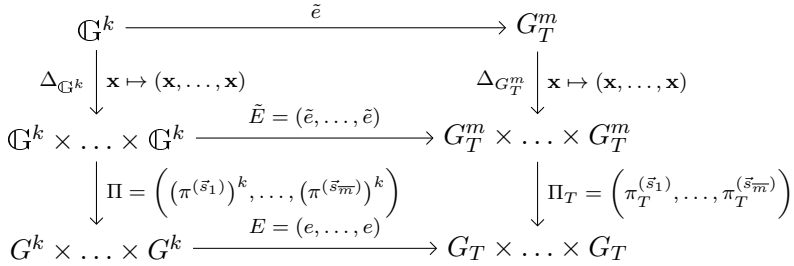

Figure 3: Fig. 2 repeated $\overline{m}$ times for an interpolating set $\vec{s}_1, \ldots \vec{s}_{\overline{m}}$ for W.

From the above Lemma 10, we have $e\Big((\pi^{(\vec{s}_i)})^k([\vec{f}_1],\ldots,[\vec{f}_k])\Big) = e\Big([f_1(\vec{s}_i)],\ldots,[f_k(\vec{s}_i)]\Big) = [(f_1 \cdots f_k)(\vec{s}_i)]_T$, where $f_j(\vec{X})$ is the polynomial defined by $f_j(\vec{X}) = \sum_t \mathfrak{q}_t(\vec{X}) f_{j,t}$. It follows that, going first down, then right in the diagram, $E(\Pi(\Delta_{\mathbb{G}^k}([\vec{f}_1],\ldots,[\vec{f}_k]))) = \big([(f_1 \cdots f_k)(\vec{s}_1)]_T,\ldots,[(f_1 \cdots f_k)(\vec{s}_{\overline{m}})]_T\big)$, from which $f_1 \cdots f_k \in W$ can be interpolated via a linear map. It follows that the span of the image of $E \circ \Pi \circ \Delta_{\mathbb{G}^k}$ has dimension at least $\overline{m} = \dim W$. But traversing the diagram first right, then down, we see that the image of $E \circ \Pi \circ \Delta_{\mathbb{G}^k}$ is contained in $(\Pi_T \circ \Delta_{G_T^m})(G_T^m)$, where $\Pi_T \circ \Delta_{G_T^m}$ is a linear map. So the dimension of the span of the image of $E \circ \Pi \circ \Delta_{\mathbb{G}^k}$ can be at most $\dim G_T^m = m$. This implies $\overline{m} \leq m$, finishing the proof.

F Applications

F.1 Instantiating Groth-Sahai Proofs

Groth-Sahai proofs are NIZK proofs of satisfiability of a set of equations in a bilinear group $\mathcal{MG}_2 := (2, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T)$. The proofs follow a basic commit-and-prove approach (for a formalisation of this see [8]) in which the witness for satisfiability (some elements in G or in $\mathbb{Z}_p$, depending on the equation type) is first committed to and then the proof shows that the committed value satisfies the equation. The common reference string includes some commitment keys which can be generated in two computationally indistinguishable ways: In the soundness setting, the keys are perfectly binding and in the witness indistinguishability setting, they are perfectly hiding. More specifically, The commitment key $[\mathbf{U}]$ is defined as $[\mathbf{U}] = [\mathbf{A}||\mathbf{A}\vec{w}]$ in the soundness setting or as $[\mathbf{U}] = [\mathbf{A}||\mathbf{A}\vec{w} - \vec{z}]$ in the witness indistinguishability setting, for $\mathbf{A} \leftarrow \mathcal{D}_\ell$, random $\vec{w}$ and public $\vec{z} \notin \mathrm{Im}(\mathbf{A})$.

In [9] it was already discussed how to instantiate Groth-Sahai proofs based on any matrix assumption, although only for the symmetric bilinear tensor product. The only point where our construction differs from the one given in ([9], Sec. 4.4) is in the definition of the symmetric bilinear map F, which we define to be $\tilde{e}$, the projecting bilinear map corresponding to polynomial multiplication associated to some matrix distribution $\mathcal{D}_\ell$, defined in Sec. 4.2. The pairing $\tilde{e}$ is described by a tuple $(\mathcal{MG}_2, [\mathbf{A}], \mathbb{G} = G^{\ell+1}, G_T^m, \tilde{e})$, $\mathbf{A} \leftarrow \mathcal{D}_\ell$ and some choice of basis for W.

The only information related to F which has to be included in the common reference string are some matrices $\mathbf{H}_1, \ldots, \mathbf{H}_\eta$ whose purpose is to ensure that, when the common reference string is defined as in the witness indistinguishability setting, the proof is correctly distributed, namely:

a) it is statistically independent of the witness (composable witness indistinguishability, [13], Def. 4),

b) a simulated proof has the same distribution of a real proof (composable zero-knowledge, [13], Def. 5).

Condition b) implies a) but for some equation types it is only possible to achieve a).

For any two vectors of elements of $\mathbb{G}$ of equal length r, $[\vec{X}] = ([\vec{x}_1], \ldots, [\vec{x}_r])$, $[\vec{Y}] = ([\vec{y}_1], \ldots, [\vec{y}_r])$ and a bilinear map $\tilde{e}$, define the map $\overset{\tilde{e}}{\bullet}$ associated with $\tilde{e}$ as $[\vec{X}] \overset{\tilde{e}}{\bullet} [\vec{Y}] = \sum_{i=1}^{r} \tilde{e}([\vec{x}_i], [\vec{y}_i])$. More specifically, depending on the equation type, the original GS proof system requires to include in the common reference string the following information which is related to $\tilde{e}$ (where $[\mathbf{U}] = [\mathbf{A}||\mathbf{A} - \vec{w}\vec{z}]$):

1) **Pairing product equations**. In this case, $\mathbf{H}_1, \ldots, \mathbf{H}_\eta$ should be a basis of the space of all matrices which are a solution of the equation $[\mathbf{UH}] \overset{\tilde{e}}{\bullet} [\mathbf{U}] = [\mathbf{0}]_T$.

2) **Multi-scalar multiplication equations**. In this case, $\mathbf{H}_1, \ldots, \mathbf{H}_\eta$ should be a basis of the space of all matrices which are a solution of the equation $[\mathbf{AH}] \overset{\tilde{e}}{\bullet} [\mathbf{U}] = [\mathbf{0}]_T$.

3) **Quadratic equations**. In this case, $\mathbf{H}_1, \ldots, \mathbf{H}_\eta$ should be a basis of the space of all matrices which are a solution of the equation $[\mathbf{AH}] \overset{\tilde{e}}{\bullet} [\mathbf{A}] = [\mathbf{0}]_T$.

Roughly speaking, the prover and the simulator use these matrices to randomize the proof, that is, they compute some proof $[\tilde{\boldsymbol{\pi}}]$ using the homomorphic porperties of the commitment scheme and then they randomize the proof as $[\boldsymbol{\pi}] = [\tilde{\boldsymbol{\pi}} + \sum r_i \mathbf{X}\mathbf{H}_i]$ (for some $\mathbf{X} \in \{\mathbf{A}, \mathbf{U}\}$ which depends on the equation type, and $r_i \leftarrow \mathbb{Z}_p$). These matrices ensure that the proof satisfies a stronger condition than a) and b) defined above, namely, that the proof is uniform conditioned on satisfying the verification equation with respect to $\tilde{e}$ (see Thm. 8, [13]).

We note that the matrices can defined independently of the choice of basis for the image space W, since a change of basis corresponds to multiplication by an invertible matrix. Therefore, these matrices can be chosen depending only on $\mathcal{D}_\ell$, without having to specify W. When $\mathcal{D}_\ell = \mathcal{L}_\ell = \mathcal{U}_\ell$, polynomial multiplication defines a map which we denote in the following by $\tilde{e}_{\mathcal{U}_\ell}$. In this case, these matrices are the same as the ones given in [9] (for the symmetric tensor product), namely matrices of the appropriate size which encode the symmetric relations which are in the kernel of $\tilde{e}_{\mathcal{U}_\ell}$. For instance, when $\ell = 2$ these matrices are:

- **Pairing product equations.** A choice of basis is:

$$\mathbf{H}_1 := \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \mathbf{H}_2 := \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \mathbf{H}_3 := \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix}.$$

- **Multi-scalar multiplication equations.** A choice of basis is:

$$\mathbf{H}_4 := \begin{pmatrix} 0 & 1 \\ -1 & 0 \\ 0 & 0 \end{pmatrix}.$$

- **Quadratic equations.** A choice of basis is:

$$\mathbf{H}_5 := \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

On the other hand, for ℓ-SCasc and the associated projecting pairing $\tilde{e}$, there are other additional relations apart from the ones derived from symmetry. For instance, for 2-SCasc and $\vec{z} = (0, 0, 1)^\top$, a possible basis of these matrices for Pairing Product Equations consists of $\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3$ as defined above plus one matrix $\mathbf{H}'$ which depends on s (such that $\mathbf{A} = \mathbf{A}(s)$) and $\vec{w}$. This dependency makes it problematic to publish $[\mathbf{H}']$. Indeed, obviously this matrix cannot be published "in the exponent", i.e. $\mathbf{H}'$ cannot be included in the common reference string. A closer look at the proofs reveals that the prover only needs $[\mathbf{U}\mathbf{H}']$, so the obvious fix would be to publish instead $[\mathbf{U}\mathbf{H}']$. However, this reveals s^2, which is sufficient to decide membership in $\mathsf{Im}\,\mathbf{A}$, i.e. to break the 2-SCasc Assumption.

For this reason, we change the specification of GS proofs. Given any symmetric projecting bilinear pairing $(\mathcal{MG}_2, [\mathbf{A}], \mathbb{G} = G^{\ell+1}, G_T^m, \tilde{e})$, the matrices $\mathbf{H}_1, \ldots, \mathbf{H}_\eta$ are defined to satisfy 1), 2) or 3) according to the equation type but with respect to $\tilde{e}_{\mathcal{U}_\ell}$. For instance, in the example of 2-SCasc for Pairing Product Equations discussed above, only $\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3$ are published. Except for this change of specification of the matrices $\mathbf{H}_1, \ldots, \mathbf{H}_\eta$, the prover, the verifier and the simulator are defined in the usual way. We briefly argue why the proof system still satisfies the required properties after this change:

- **Completeness.** If $\mathbf{H}$ is such that $[\mathbf{X}\mathbf{H}] \overset{\tilde{e}_{\mathcal{U}_\ell}}{\bullet} [\mathbf{Y}] = [\mathbf{0}]_T$, for some $\mathbf{X}, \mathbf{Y} \in \{\mathbf{A}, \mathbf{U}\}$ then it follows from the symmetry of $\tilde{e}$ that it also holds that $[\mathbf{X}\mathbf{H}] \overset{\tilde{e}}{\bullet} [\mathbf{Y}] = [\mathbf{0}]_T$. Roughly speaking, this means that the randomization of the proof cancels out not only with $\tilde{e}_{\mathcal{U}_\ell}$ but also with $\tilde{e}$. More formally, this implies that the proof of completeness from [13], Thm. 6 can be used unchanged to prove completeness in this case.

- **Soundness.** It follows from [13], Thm. 7 that soundness depends only on the projecting property of the map $\tilde{e}$.

- **Witness Indistinguishability/ Zero-Knowledge.** Both a real proof and a simulated proof have the same distribution, namely the uniform one conditioned on the verification equation being satisfied *with respect to* $\tilde{e}_{\mathcal{U}_\ell}$.

Efficiency discussion. As mentioned, the important efficiency measures for GS proofs are common reference string size, proof size, prover's and verification's efficiency. Proof size depends only on the size of the matrix assumption and the equation type, so it's omitted here. Essentially, so does the efficiency of the prover, with some minor differences which are discussed below. For the rest, the discussion goes as follows:

- **Size of the common reference string.** ℓ-SCasc assumption is the most advantageous from this point of view (as noted in [9]) since the commitment keys can be described more compactly in this case.

- **Prover's computation.** The cost of computing a commitment depends roughly on the sparseness of the matrix $\mathbf{A}$. For instance, for the uniform assumption with $\ell = 2$, a commitment costs at least 6 exponentiations, so Lin2 and 2-SCasc are more advantageous. Following [8], the prover's computation can be reduced significantly by allowing a prover to choose its own common reference string and then proving its correctness. This allows to minimize the number of exponentiations, since the prover knows $s \in \mathbb{Z}_p$ and can compute most operations in $\mathbb{Z}_p$. Obviously the same trick applies here.
- **Verification cost.** Verification cost is the efficiency measure which depends more on the choice of pairing, as it typically involves several evaluations of the map F. Since the map $\tilde{e}$ can be computed more efficiently than F, the verification cost is significantly reduced for many equation types. For instance, using our map derived from 2-SCasc we can save 4 basic pairing evaluations per evaluation of $\tilde{e}$.

We emphasize that this discussion is for general equation types. For some specific types — like linear equations with constants in G, the new map does not imply more efficient verification.

We conclude that the 2-SCasc instantiation with polynomial multiplication is definitely the most efficient implementation for GS NIZK proofs in symmetric bilinear groups, not only because of the size of the common reference string as pointed out in [9] but also from point of view of efficiency of verification.

F.2 Efficient Implementation of the k-times Homomorphic BGN Cryptosystem

In this section we show how to implement a multilinear variant of the Boneh-Goh-Nissim cryptosystem from [3] with prime-order multilinear groups. We proceed as follows: first we transform a given prime-order multilinear group into a projecting composite-order multilinear group using the results from Sec. 4.2. As seen in Sec. 5.2, the most efficient way to do this is using the k-SCasc assumption. We write the generator, already given in Ex. 1, again in more detail. In the next step, we show how to implement the BGN cryptosystem in those groups and compare the implementation costs to implementations of k-BGN derived from the work of Freeman ([10]) and Seo ([21]).

Example 4 (A generator for the BGN cryptosystem)**.** *Let $k \in \mathbb{N}$ and $\mathcal{SC}_k$ denote the matrix distribution belonging to the symmetric cascade assumption from Def. 2. Let $\mathcal{G}_{k-BGN}$ be an algorithm that, on input a security parameter $\lambda \in \mathbb{N}$, does the following:*

- *Obtain $\mathcal{MG}_k := (k, G, G_T, e, p, \mathcal{P}, \mathcal{P}_T)$ from a symmetric k-linear group generator $\mathcal{G}_k(\lambda)$.*
- *Let $\mathbb{G} := G^{k+1}, \mathbb{G}_T := G_T^{k^2+1}$*
- *Choose $s \xleftarrow{R} \mathbb{Z}_p$ and let $\mathbb{H}^{(s)} \subset \mathbb{G}, \mathbb{H}_T^{(s)} \subset \mathbb{G}_T$ as in Sec. 4.1.*
- *Let $\tilde{e}([\vec{f_1}], \ldots, [\vec{f_k}]) := [f_1 \cdots f_k]_T$*
- *Output the tuple $(\mathcal{MG}_k, \mathbb{H}^{(s)}, \mathbb{G}, \mathbb{G}_T, \tilde{e})$.*

Observe that $\mathcal{G}_{k-BGN}$ is a $(k, \mathcal{SC}_k)$ multilinear map generator. Additionally, $\mathcal{G}_{k-BGN}$ is projecting for $\mathbb{H}^{(s)}$ w.r.t. the maps $\pi\colon \mathbb{G} \to G$, $[\vec{f}] \mapsto [f(s)]$ and $\pi'\colon \mathbb{G}_T \to G_T$, $[\vec{f}]_T \mapsto [f(s)]_T$. Both maps can be computed efficiently given s. From the discussion in Sec. 4.1 it follows that if $\mathcal{G}_k$ satisfies the k-SCasc assumption, then $\mathcal{G}_{k-BGN}$ satisfies the subgroup indistinguishability property. As seen in Sec. 4.2, the computation of the map $\tilde{e}$ can be optimized using techniques for fast polynomial multiplication.

The additively and one time multiplicatively homomorphic BGN encryption scheme from [3] uses a pairing and can be extended in a straightforward way to work with a k-linear map for arbitrary $k \in \mathbb{N}$. The resulting encryption scheme is then $k-1$ times multiplicatively homomorphic. We now describe how to implement the scheme using $\mathcal{G}_{k-BGN}$.

Setup(1^λ): Run $\mathcal{G}_{k-BGN}$ to obtain $(\mathcal{MG}_k, \mathbb{H}^{(s)}, \mathbb{G}, \mathbb{G}_T, \tilde{e})$. Output $\mathcal{PK} := (p, \mathbb{G}, \mathbb{G}_T, \tilde{e}, \mathcal{P}, [s])$ and $\mathcal{SK} := s$

Enc$(\mathcal{PK}, m)$: To encrypt a message in $\mathbb{G}$, draw $h_0, \ldots, h_k \xleftarrow{R} \mathbb{Z}_p \setminus \{0\}$ and compute $h := [(-sh_0, h_0 - sh_1, \ldots, h_{k-1} - sh_k, h_k)] \in \mathbb{H}^{(s)}$ using $[s]$ from $\mathcal{PK}$. Set $\mathcal{P}^0 := [(1, 0, \ldots, 0)]$. Compute and output the ciphertext as

$$c := m \cdot \mathcal{P}^0 + h = [(-sh_0 + m, h_0 - sh_1, \ldots, h_{k-1} - sh_k, h_k)]$$

Encryption in $\mathbb{G}_T$ works similarly.

Generator	ciphertexts (in $\mathbb{G}/\mathbb{G}_T$)		Enc / Dec (in $\mathbb{G}/\mathbb{G}_T$)		Mult	
	(el. from G)	(el. from G_T)	(exp. in G)	(exp. in G_T)	(exp. in G_T)	(eval. of e)
Freeman, symm.	$k+1$	$(k+1)^k$	$(k+1)^2$	$(k+1)^{2k}$	-	$(k+1)^{k+1}$
Freeman, asymm.	$2k$	$2^k \cdot k$	2^2	2^{2k}	-	2^k
Seo, symmetric	$k+1$	$\binom{2k}{k}$	$(k+1)^2$	$\binom{2k}{k}^2$	-	$(k+1)^k$
This paper, $\mathcal{G}_{k-BGN}$	$k+1$	k^2+1	$k+1$	k^2+1	-	$(k+1)^k$
This paper, $\mathcal{G}_{k-BGN}$, opt.	$k+1$	k^2+1	$k+1$	k^2+1	$(k^3+k)k$	k^2+1

Table 2: Implementation costs and ciphertext sizes of k-BGN with the generators obtained by extending the construction of Freeman ([10]), the generator $\mathcal{G}_{k,\mathcal{U}_k}$ used by Seo ([21]) and our most efficient generator $\mathcal{G}_{k-BGN}$. The latter is listed twice, differing in the method to compute the mapping $\tilde{e}$ (naively or optimized using techniques for fast polynomial multiplication.) Costs are stated in terms of application of the basic map e and exponentiations in G, respectively G_T. To keep the exposition simple, we measure ciphertext sizes of the coefficient representation (not the optimized point-value representation introduced in Sec. 4). Observe that our construction is the only one for which tradeoffs between basic multilinear map evaluations and exponentiations are known.

Dec$(\mathcal{SK}, c)$: Decryption in $\mathbb{G}$ and $\mathbb{G}_T$ works by applying π, i.e. evaluating c, interpreted as polynomial $c(X)$, in $\mathcal{SK} = s$. For this, parse $c := (c_1, \ldots, c_\ell)$ and compute $\pi(c) = [c(s)] = [c_1 + s \cdot c_2 + \cdots + s^\ell \cdot c_\ell]$. Output $m = \log_{\mathcal{P}}(c(s))$.

Add(PK, c, c'): We assume $c, c' \in \mathbb{G}$. Draw $\hat{h} \xleftarrow{R} \mathbb{H}^{(s)}$. Compute and output

$$c + c' + \hat{h} = (m + m') \cdot \mathcal{P}^0 + h + h' + \hat{h}$$

Adding encrypted messages in $\mathbb{G}_T$ works just as in $\mathbb{G}$.

Mult$(\mathcal{PK}, c_1, \ldots, c_k)$: We require $c_1, \ldots, c_k \in \mathbb{G}$. Draw $\hat{h} \xleftarrow{R} \mathbb{H}_T^{(s)}$. Compute and output

$$\tilde{e}(c_1, \ldots, c_k) + \hat{h} = (m_1 \cdot \ldots \cdot m_k)\mathcal{P}_T^0 + \tilde{h} + \hat{h}$$

where $\mathcal{P}_T^0 := [(1, 0, \ldots, 0)]_T$ and $\tilde{h} \in \mathbb{H}^{(s)}$.

Observe that correctness of decryption follows from $c(s) = [-sh_0 + m + s(h_0 - sh_1) + \cdots + s^{k-1}(h_{k-1} - sh_k) + s^k(h_k)] = [m + h(s)] = [m]$. The number of G-exponentiations required for encryption and decryption is equal to the number of copies of G used for $\mathbb{G}$ and $\mathbb{G}_T$, i.e. $k+1$ for $\mathbb{G}$ and k^2+1 for $\mathbb{G}_T$.

Corollary 1. *The above scheme is semantically secure if the group generator $\mathcal{G}_k$ satisfies the k-*SCasc *assumption.*

Proof. Semantic security follows from a straightforward adaption of Theorem 3.1 from [3] and the fact that $\mathcal{G}_{k-BGN}$ satisfies the subgroup indistinguishability property if $\mathcal{G}_k$ satisfies the k-SCasc assumption. □

Comparison to an extension of Freeman's construction ([10]). The projecting pairing from [10] has a natural extension to the multilinear case. For $k \in \mathbb{N}$, the k-linear extension of the symmetric bilinear generator of [10], Theorem 2.5, is a $(k, \mathcal{U}_k)$ multilinear map generator (note that Freeman uses the uniform distribution to generate subgroups). We can define the k-linear map such that it is projecting, following [10], Section 3.1 (using the notation from the original paper). Thus, we let the generator compute $\tilde{e}$ as $\tilde{e}([\vec{f}_1], \ldots, [\vec{f}_k]) := e(\mathcal{P}, \ldots, \mathcal{P})^{\vec{f}_1 \otimes \cdots \otimes \vec{f}_k}$. This setting can be further optimized for multilinear maps if we use an asymmetric prime-order map as a starting point for an asymmetric generator. We will not go into the details, since the construction is essentially the same as in [10], Example 3.3, naturally extended to the multilinear setting. On a high level, the main advantage is that we can keep the dimension of the subgroups and thus the composite-order groups small (i.e., $\mathbb{G}_i := G_i^2$), leading to a smaller (though still exponentially large) number of basic multilinear map evaluations to compute $\tilde{e}$ (cf. Tab. 2). Note that even the asymmetric generator, using $\mathbb{G}_i = G_i^2$, requires $2k$ group elements to describe ciphertexts in the base groups. This is because the BGN cryptosystem has to be adjusted to work with an asymmetric map (see [10], Section 5, for details).

Comparison to an extension of Seo's construction ([21]). As we explained in Sec. 3, in the constructions of Freeman, subgroups are always sampled according to the uniform assumption. Under this condition, Seo ([21]) proved that Freeman's construction of a projecting pairing is not optimal in the symmetric case. For this case, Seo gives a projecting pairing that is optimal in Freeman's model and, as seen in Appendix G, matches our construction for the uniform assumption, and can therefore be generalized to the multilinear case (see Ex. 3). The implementation costs of k-BGN using Seo's construction can be seen in Tab. 2.

G A Unified View on Different Projecting Pairings From the Literature

In this section, we compare our constructions for the special case of a 2-linear map with previous constructions of Groth and Sahai[10] ([13]) and Seo ([21]). We use the language of Seo to represent all constructions consistently. Let us first briefly introduce the required tools for this.

Given two vectors $\vec{x} = (x_0, \ldots, x_{n-1}) \in \mathbb{Z}_p^n$ and $\vec{y} = (y_0, \ldots, y_{n-1}) \in \mathbb{Z}_p^n$, the tensor product $\vec{x} \otimes \vec{y}$ is defined as $(x_0y_0, x_0y_1, x_0y_2, \ldots, x_{n-1}y_{n-1})$. Any bilinear map $\tilde{e}\colon G^n \times G^n \to G_T^m$ can be uniquely described by a matrix $\mathbf{B} \in \mathbb{Z}_p^{n^2 \times m}$ such that $\tilde{e}([\vec{x}], [\vec{y}]) = e([1], [1])^{(\vec{x}\otimes\vec{y})\ \mathbf{B}} = [(\vec{x} \otimes \vec{y})\ \mathbf{B}]_T$.

We can now present the pairing $\tilde{e}$ of each construction in terms of the matrix $\mathbf{B}$.

1. Symmetric tensor product (original Groth-Sahai construction)

$$\mathbf{B} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

2. Seo's construction

$$\mathbf{B} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \in \mathbb{Z}_p^{9\times 6}.$$

Seo's construction can be also written as: $\tilde{e}([\vec{x}], [\vec{y}]) := [(x_0y_0, x_0y_1 + x_1y_0, x_0y_2 + x_2y_0, x_1y_1, x_1y_2 + x_2y_1, x_2y_2)]_T$. Seo proves that his construction is projecting for the $\mathcal{U}_2$ Assumption. We note that our construction for 2-Lin and $\mathcal{U}_2$ is exactly the same if we choose as a basis for W the set $\{\mathfrak{q}_i\mathfrak{q}_j : 0 \le i \le j \le 2\}$, for the polynomials $\mathfrak{q}$ defined in example 2 and 3, respectively.

[10] The most efficient symmetric construction of Freeman ([10]), based on 2-Lin, matches the one of Groth and Sahai and is thus not listed here.

3. Our construction for the $\mathcal{SC}_2$ assumption, choosing $\{1, X, X^2, X^3, X^4\}$ as a basis for W.

$$\mathbf{B} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \in \mathbb{Z}_p^{9\times 5}$$

Our construction can be also be written as
$\tilde{e}([\vec{x}], [\vec{y}]) := [(x_0y_0, x_0y_1 + x_1y_0, x_0y_2 + x_2y_0 + x_1y_1, x_1y_2 + x_2y_1, x_2y_2)]_T$.

4. Our construction for the $\mathcal{SC}_2$ assumption with an alternative choice for the basis of W:

$$\mathbf{B} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{T} \in \mathbb{Z}_p^{9\times 5}, \quad \text{where} \quad \mathbf{T} := \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ -2 & -1 & 0 & 1 & 2 \\ 4 & 1 & 0 & 1 & 4 \\ -8 & -1 & 0 & 1 & 8 \\ 16 & 1 & 0 & 1 & 16 \end{pmatrix}.$$

This construction can be also be written as

$$\tilde{e}([\vec{x}], [\vec{y}]) := \Big[\Big(\sum_{t=0}^{4}\sum_{i+j=t} x_iy_j(-2)^t, \sum_{t=0}^{4}\sum_{i+j=t} x_iy_j(-1)^t, x_0y_0, \sum_{t=0}^{4}\sum_{i+j=t} x_iy_j, \sum_{t=1}^{4}\sum_{i+j=t} x_iy_j2^t\Big)\Big]_T$$

G.1 Efficiency Improvement for Seo's Construction

Seo claims that his pairing (item (2) above) is optimal among all based on the uniform subgroup decision assumption in terms of a) image size and b) number of basic pairing operations. Regarding a), our results do not contradict Seo's claim. Regarding b), Seo claims that the number of basic pairing operations is at least the weight of the matrix $\mathbf{B}$, which is 9 for the $\mathcal{U}_2$ Assumption.

Seo's implicit assumption behind this seems to be that if the pairing has the form $(\vec{x} \otimes \vec{y})\mathbf{B}$ for some matrix $\mathbf{B}$, then the best way to compute it is also via such a vector-matrix product: for each non-zero entry $B_{i,j}$ of B, compute $B_{i,j}x_iy_j$ and then perform some additions in the exponent. This then corresponds to one pairing operation (computing products of x_i and y_j in the exponent, $B_{i,j}$ is a scalar) per non-zero entry of $\mathbf{B}$. We reduce this to the rank of $\mathbf{B}$ by applying linear transformations to $\vec{x}$, $\vec{y}$ prior to multiplication (more precisely, treating $\vec{x}$ and $\vec{y}$ as polynomials and interpolating).

More formally, for any pairing $\tilde{e}$ with associated matrix $\mathbf{B}$, to reduce the number of basic pairing operations to m, it suffices to find matrices $\mathbf{C} \in \mathbb{Z}_p^{(\ell+1)\times m}, \mathbf{D} \in \mathbb{Z}_p^{m\times m}$ such that

$$\tilde{e}([\vec{x}], [\vec{y}]) := [(\vec{x} \otimes \vec{y})\ \mathbf{B}]_T = [((\vec{x}\mathbf{C}) \otimes (\vec{y}\mathbf{C})) \begin{pmatrix} \mathbf{I}_m \\ \mathbf{0}_{(m^2-m)\times m} \end{pmatrix} \mathbf{D}]_T.$$

Given these matrices, the pairing $\tilde{e}$ can be computed with only m evaluations of e, regardless of the weight of $\mathbf{B}$, as follows:

1. Compute $[\vec{u}] = [\vec{x}\mathbf{C}] \in G^m$ and $[\vec{y}] := [\vec{y}\mathbf{C}] \in G^m$.
2. Compute $[\vec{w}]_T = (e([u_1],[v_1]),\ldots,e([u_m],[v_m])) = ([u_1v_1]_T,\ldots,[u_mv_m]_T) = [(\vec{u}\otimes\vec{v})\begin{pmatrix}\mathbf{I}_m\\ \mathbf{0}_{m^2-m}\end{pmatrix}]_T$.
3. Compute the final $[\vec{z}]_T$ as $[\vec{z}]_T = [\vec{w}\mathbf{D}]_T$.

Note that steps 1,3 require only group operations in G, G_T.

In the specific case of Seo's construction (item (2) in the list) — which matches our construction for $\mathcal{U}_2$ for the basis $\{\mathfrak{q}_i\mathfrak{q}_j \mid 0 \le i \le j \le 2\}$ of W —, $m = 6$ and the matrices $\mathbf{C} \in \mathbb{Z}_p^{3\times 6}, \mathbf{D} \in \mathbb{Z}_p^{6\times 6}$ are defined as:

$$\mathbf{C} := \begin{pmatrix}\mathfrak{q}_0(\vec{x}_1) & \ldots & \mathfrak{q}_0(\vec{x}_6)\\ \mathfrak{q}_1(\vec{x}_1) & \ldots & \mathfrak{q}_1(\vec{x}_6)\\ \mathfrak{q}_2(\vec{x}_1) & \ldots & \mathfrak{q}_2(\vec{x}_6)\end{pmatrix}, \qquad \mathbf{D} := \begin{pmatrix}(\mathfrak{q}_0\cdot\mathfrak{q}_0)(\vec{x}_1) & \ldots & (\mathfrak{q}_0\cdot\mathfrak{q}_0)(\vec{x}_6)\\ (\mathfrak{q}_0\cdot\mathfrak{q}_1)(\vec{x}_1) & \ldots & (\mathfrak{q}_0\cdot\mathfrak{q}_1)(\vec{x}_6)\\ (\mathfrak{q}_0\cdot\mathfrak{q}_2)(\vec{x}_1) & \ldots & (\mathfrak{q}_0\cdot\mathfrak{q}_2)(\vec{x}_6)\\ (\mathfrak{q}_1\cdot\mathfrak{q}_1)(\vec{x}_1) & \ldots & (\mathfrak{q}_1\cdot\mathfrak{q}_1)(\vec{x}_6)\\ (\mathfrak{q}_1\cdot\mathfrak{q}_2)(\vec{x}_1) & \ldots & (\mathfrak{q}_1\cdot\mathfrak{q}_2)(\vec{x}_6)\\ (\mathfrak{q}_2\cdot\mathfrak{q}_2)(\vec{x}_1) & \ldots & (\mathfrak{q}_2\cdot\mathfrak{q}_2)(\vec{x}_6)\end{pmatrix}^{-1},$$

where $\mathfrak{q}_0(\vec{X}) = X_{21}X_{32} - X_{22}X_{32}$, $\mathfrak{q}_1(\vec{X}) = X_{11}X_{32} - X_{12}X_{31}$, $\mathfrak{q}_2(\vec{X}) = X_{11}X_{22} - X_{12}X_{21}$ and $\vec{x_i} \in \mathbb{Z}_p^6$ are any interpolating set for the space spanned by $\{\mathfrak{q}_i\mathfrak{q}_j \mid 0 \le i \le j \le 2\}$, which guarantees that $\mathbf{D}$ is properly defined. This allows us to bring down the number of basic pairing operations to only 6 instead of 9, which was the number of operations which Seo claims to be necessary for compute $\tilde{e}$.

Note that by changing the choice of basis for W we can also get an even more efficient projecting pairing for the uniform assumption. In the language we just introduced, this amounts to choose $\mathbf{C}$ as above but define $\mathbf{D}$ as the identity matrix. This allows us to save all the exponentiations in $\mathbb{G}_T$.

H Implementation with Multilinear Map Candidates

H.1 The Candidate Multilinear Maps from [11, 6]

In this section, we investigate to what extent our constructions can be implemented with the recent candidates [11, 6] of approximate multilinear maps. These works only provide approximations of multilinear maps in the following sense. Namely, instead of group elements, [11, 6] define "noisy encodings." Essentially, a noisy encoding is a group element with an additional noise term. This means that there is a whole set of encodings $\mathsf{Enc}(g)$ of a group element g. Each operation on encodings increases the size of their noise terms. (More specifically, the noise term of the result of an operation is larger than the noise terms of the inputs.) In particular, each encoding can be used only for an a-priori limited number of operations. After that, its noise term becomes too large, and errors in computations may occur.

This noisy encoding of group elements has a number of effects which are relevant for our constructions:

Group membership hard to decide. It is not efficiently decidable whether a given encoding actually encodes *any* group element (with a certain noise bound).

Non-trivial comparisons. To determine whether two given encodings encode the same group element (i.e., lie in the same set $\mathsf{Enc}(g)$), we require a special comparison algorithm (which however can be made publicly available).

Non-unique computations. Even if two computations yield encodings of the same group element, the actual encodings may differ. Specifically, an encoding may leak (through its noise term) the sequence of operations used to construct it. To hide the sequence of performed operations, there exists a re-randomization algorithm that re-randomizes the noise term (essentially by adding a substantially larger noise term).

Black-box exponents. It is possible to choose (almost) uniformly distributed exponents, but these can only be used in a black-box way (using addition, subtraction, and multiplication), and without using their explicit integer representation.

Subgroup membership problems. The construction in [11] allows for a very generic attack on subgroup membership assumptions in the (encoded) "source group" of the multilinear map. In particular, matrix assumptions like SCasc or the ℓ-linear assumption do not appear to hold in the source group. On the other hand, the construction in [6] does support subgroup membership assumptions (and in particular matrix assumptions) in the source group.

H.2 Our Constructions with the Multilinear Map Candidates

We now inspect our constructions for compatibility with approximate multilinear maps as sketched above. Syntactically, our constructions (from Sec. 4.2 and Appendix B) start from a given group G and a k-linear map $e\colon G^k \to G_T$, and construct another group $\mathbb{G} = G^n$, along with $\mathbb{G}_T = G_T^m$ and a k-linear map $\tilde{e}\colon \mathbb{G}^k \to \mathbb{G}_T$. In both cases, computations in $\mathbb{G}, \mathbb{G}_T$, and the evaluation of $\tilde{e}$ can be reduced to computations in G, G_T, and evaluating e. Hence, at least syntactically, our constructions can be implemented also with approximate multilinear maps as above. But of course, this does not mean that our constructions also retain the security properties we have proved when implemented in an approximate setting. Hence, we now investigate the effect of the imperfections sketched above.

Group membership hard to decide. We have assumed that group membership in our constructed group $\mathbb{G}$ is easy to decide. This of course no longer holds if the underlying prime-order group cannot be efficiently decided in the first place. We stress that this has no implications on *our* results, but of course makes the constructed group $\mathbb{G}$ also less useful in applications.

Non-trivial comparisons. Since, in our constructions, we never explicitly use comparisons, we also never need to use a comparison algorithm. On the other hand, a comparison in the groups $\mathbb{G}$ and $\mathbb{G}_T$ we construct can be reduced to comparing elements of G and G_T.

Non-unique computations. In the (encoded) groups we construct, the noise of the underlying G- or G_T-elements also leaks information about the performed computations. However, this noise can be re-randomized by re-randomizing the noise of the underlying G- and G_T-elements.

Black-box exponents. Both of our constructions use exponents only in a black-box way. Specifically, exponents are only uniformly chosen, added, and multiplied both during setup and operation of the scheme. (One subtlety here is the computation of the "reduced polynomials" $[h_i] = [X^i \bmod h]$ in the projecting and canceling construction from Appendix B. Note that the coefficients of these h_i can be computed from the coefficients of h through linear operations alone. Hence, the involved exponents do not have to be explicitly divided.) However, since we assume (special types of) matrix assumptions in the source group, we can only use the candidate of [6] for our constructions.

Summarizing, both of our constructions can be implemented with the approximate multilinear map candidate from [6] (but not with the one from [11]).