Trapdoor one-way functions from tensors

Anand Kumar Narayanan

anandkumar.n@gmail.com

Abstract. Update: We retract the claim that main construction in the paper is cryptographically secure. In particular, the trapdoor functions generated in Algorithm 1 from three dimensional tensors are efficiently distinguishable from random tensors, since their two dimensional slices (in the longest dimension) are rank deficient. We leave the article as a record, in hopes that some of the techniques are useful in other contexts.

Weyman and Zelevinsky generalised Vandermonde matrices to higher dimensions, which we call Vandermonde-Weyman-Zelevinsky tensors [37]. We generalise Lagrange interpolation to higher dimensions by devising a nearly linear time algorithm that given a Vandermonde-Weyman-Zelevinsky tensor and a sparse target vector, finds a tuple of vectors that hit the target under tensor evaluation. Tensor evaluation to us means evaluating the usual multilinear form associated with the tensor in all but one chosen dimension. Yet, this interpolation problem phrased with respect to a random tensor appears to be a hard multilinear system. Leveraging this dichotomy, we propose preimage sampleable trapdoor one-way functions in the spirit of Gentry-Peikert-Vaikuntanathan (GPV) lattice trapdoors [17]. We design and analyse "Hash-and-Sign" digital signatures from such trapdoor one-way functions, yielding short signatures whose lengths scale nearly linearly in the security parameter. We also describe an encryption scheme.

Our trapdoor is a random Vandermonde-Weyman-Zelevinsky tensor over a finite field and a random basis change. We hide the Vandermonde-Weyman-Zelevinsky tensor under the basis change and publish the resulting pseudorandom tensor. The one way function is the tensor evaluation derived from the public tensor, restricted so as to only map to sparse vectors. We then design the domain sampler and preimage sampler demanded by the GPV framework. The former samples inputs that map to uniform images under the one-way function. The latter samples preimages given supplementary knowledge of the trapdoor. Preimage sampling is a randomised version of interpolation and knowing the basis change allows efficient translation between interpolation corresponding to the public and trapdoor tensors. An adversary seeking a preimage must solve a pseudorandom multilinear system, which seems cryptographically hard.

1 Introduction

1.1 Cryptographic context

Tensor based cryptography. Most linear algebraic problems such as solving linear systems, computing eigenvalues or rank of a matrix, telling if two matrices are equivalent/similar/congruent etc. are computationally easy. Jumping from two to three dimensions or more, their multilinear algebraic analogues concerning tensors become hard [21,19]. Phase transition in hardness from two to three dimensions is seen elsewhere, such as going from 2-SAT to 3-SAT, or 2-COLOR to 3-COLOR. Unlike 3-SAT or 3-COLOR, some three or higher dimensional multilinear algebraic tensor problems offer average case hardness suited for cryptography. Tensor isomorphism problems are a cryptographically significant such family, generalising matrix equivalence/similarity/congruence to higher dimensional tensors [19,8,20]. Its canonical representative is the eponymous Tensor Isomorphism (**TI**) problem: given two tensors over a finite field, decide if they are isomorphic. Two tensors are isomorphic if there is a tuple of invertible square matrices that takes one tensor to the other, by multiplication in the respective dimensions. The search version of **TI** asks for an isomorphism, which is a tuple of matrices. There is a close knit family **TI**-complete of hard problems with tight (linear or quadratic time) reductions between each other. This family includes well studied multivariate cryptographic problems (restricting to symmetric tensors) [28] and long standing group theoretic problems like p-group isomorphism [35]. With a notion of hard to compute isomorphism at hand, it is hard not to derive a zero-knowledge identification scheme using the Goldreich-Micali-Wigderson motif [18], which under the Fiat-Shamir transformation yields digital signatures [12]. Since only polynomial factor quantum speedups are known for **TI**-complete problems, the digital signatures that are based on **TI**-complete problems are considered post-quantum secure. Digital signature schemes based on tensor isomorphism problems such as ALTEQ [7,36] and MEDS [9] were part of the first round of NIST's recent on-ramp competition. MEDS is based on **TI** in three dimensions and ALTEQ is based on **TI** restricted to alternating tensors with the same matrix acting in each dimension. Curiously, neither ALTEQ nor MEDS made it to NIST's second round, perhaps due to longer signatures and keys in comparison to the competition. Many of the on-ramp signature scheme selected for NIST's second round are based on the Multi-Party Computation in the Head (MPCitH) paradigm, with signature lengths scaling quadratically in the security parameter [4].

Trapdoors. Informally, trapdoor one-functions are a family of pairs of functions and trapdoors. The function is easy to compute, but hard to invert. Unless, presented with the corresponding trapdoor as additional information, when the function becomes easy to invert. Trapdoor one-way function constructions are rare in general. The best known construction is perhaps the one underlying RSA. The function is modular exponentiation modulo a composite (say, an RSA

number: a product of two large odd prime numbers) and the corresponding trapdoor is the prime factorisation of the composite. A closely related construction is due to Rabin [30], where the function is squaring modulo a composite and the corresponding trapdoor is again the factorisation of the composite. Curiously, it is not known if there are trapdoors for computing discrete logarithms (neither in multiplicative groups of finite fields nor in elliptic curve groups over finite fields), a testament to how special trapdoors are. Other important trapdoor constructions come from code based and lattice based cryptography. Code based trapdoor constructions come in two flavours. The first is a McEliece [24] style trapdoor: a description of the structure an error correction code the facilitates efficient decoding. This structure is hidden through a basis transformation to derive a public generator matrix for the same code which hides the structure. The one-way function then is encoding function of the code, based on the public generator matrix. The second flavour was pioneered by Alekhnovich [5], who devised a trapdoor to decode random linear codes. This line of work was specialised to random quasi-cyclic codes culminated in HQC, the latest addition to NIST's suit of post-quantum encryption schemes [1]. Lattice trapdoors were pioneered by Ajtai [2,3], whose worst case to average case reductions laid the complexity theoretic foundations of lattice based cryptography. A second revolution started with the Gentry, Peikert and Vaikuntanathan (GPV) construction of preimage sampleable lattice trapdoor functions [17]. The carefully curated randomness in their preimage samples was critical in eluding cryptanalytic attacks [27] (that had completely broken previous attempts at lattice trapdoors) and in rigorously proving the security of the resulting Hash-and-Sign signature schemes. The latter, specialised to algebraically structure lattices in certain cyclotomic rings resulted in FALCON [29], one of two NIST approved lattice based post-quantum signatures. Micciancio and Peikert extended, simplified and optimised the constructions of GPV, which we refer to for a more comprehensive history of lattice based trapdoors [25].

Preimage sampleable trapdoors one-way functions. We follow and meet the GPV definition of preimage sampleable trapdoor functions, but our construction has little to do with theirs. GPV define a preimage sampleable function family, through three probabilistic polynomial time algorithms: a trapdoor generator, a domain sampler and a preimage sampler. The trapdoor generator takes in a desired security parameter and outputs a domain, a range, a function mapping from that domain to the range, and a trapdoor. The domain sampler takes the function description as input and samples from the domain such that the image of the sample under the function is uniform. The sample itself, does not have to be uniform in the domain. The preimage sampler takes the function description, the trapdoor and a target image as input and samples from the domain such that the image of the sample hits the target. Knowing only function and not the trapdoor, it should be cryptographically hard to compute preimages. Some applications demand collision resistance.

1.2 Contributions

We construct the first trapdoor one-way functions from tensors over finite fields. We use them to design Hash-and-Sign signatures with signature lengths scaling as $\mathcal{O}(\lambda \log \lambda)$ in the security parameter λ. We anticipate that our trapdoors from tensors bring tensor based cryptography closer to being an alternative for lattice and code based post-quantum cryptography, both in terms of performance and the applicability to advanced cryptographic primitives that need trapdoors. To further bridge the gap, our public key length, which remains cubic in λ, needs to be lowered. A first impression of the construction and its cryptographic application follows.

Boundary and doubly boundary format tensors. We work exclusively with tensors of forms called boundary formats. Boundary formats are the ones that generalise the notion of square matrices to higher dimensions, in the strictest sense according to the theory of hyperdeterminants [15,16]. Let us permute the dimensions if needed to make the first dimension the longest. Here on, the first dimension, which is the one where the tensor is the longest will be special. Then, boundary formats in three dimensions are of the form $(k_2 + k_3 + 1) \times (k_2 + 1) \times (k_3 + 1)$. The projective length (one fewer than the format length) $k_2 + k_3$ in the first dimension equals the sum of those in the other two. Likewise, in arbitrary dimensions boundary formats are those where the projective length in the first dimension equals the sum of the projective lengths in the other dimensions. We call a boundary format as doubly boundary if it remains a boundary format even after removing the first dimension (permuting the remaining dimensions, if needed). Doubly boundary formats in three dimensions are all of the form $(2k + 1) \times (k + 1) \times (k + 1)$, which will be the stage where we will rigorously analyse our constructions. The four dimensional doubly boundary formats $(4k + 1) \times (2k + 1) \times (k + 1) \times (k + 1)$ will also play an important role, for applications that require sampling from a large preimage distribution. In summary, we base our constructions on generalisations of square matrices such as boundary formats $(2k+1) \times (k+1) \times (k+1)$ and $(4k+1) \times (2k+1) \times (k+1) \times (k+1)$, as opposed to the more familiar cubical $k \times k \times k$ formats. Despite drawing inspiration from the theory of hyperdeterminants, our constructions are completely elementary and require no knowledge of the theory. Being in boundary formats will make sure our constructions have the right degrees of freedom to meet certain constraints, but not more for an adversary to exploit. In general, a boundary format is one of the form $(k_1 + 1) \times (k_2 + 1) \times \ldots \times (k_r + 1)$ with $k_1 = k_2 + k_3 + \ldots + k_r$. A doubly boundary format satisfies the extra constraint $k_2 = k_3 + k_4 + \ldots + k_r$. We have implicitly assumed that the two longest dimensions are the first and the second, without loss of generality. Since the tensor isomorphism problem is best studied in the three dimensional case, we initially recommend these constructions in three or a few more dimensions. Further, it is unclear if there is benefit at all to go to much higher dimensions, since the description length of the tensors will grow exponentially with dimension r. Throughout, it is good to keep in mind

the doubly boundary three dimensional format $(2k+1) \times (k+1) \times (k+1)$, as the recurring example in our descriptions.

1.3 Vandermonde-Weyman-Zelevinsky tensors

A column vector determines a (say, square) Vandermonde matrix. Analogously, given a desired boundary format, Weyman and Zelevinsky take $r-1$ column vectors to determine a structured tensor in the format which generalises Vandermonde matrices [37, Sec. 7]. We will call such tensors as Vandermonde-Weyman-Zelevinsky tensors. The smallest example that takes two 3×1 vectors to get $3 \times 2 \times 2$ tensors is illustrated in figure 1.3 It is convenient to package the two

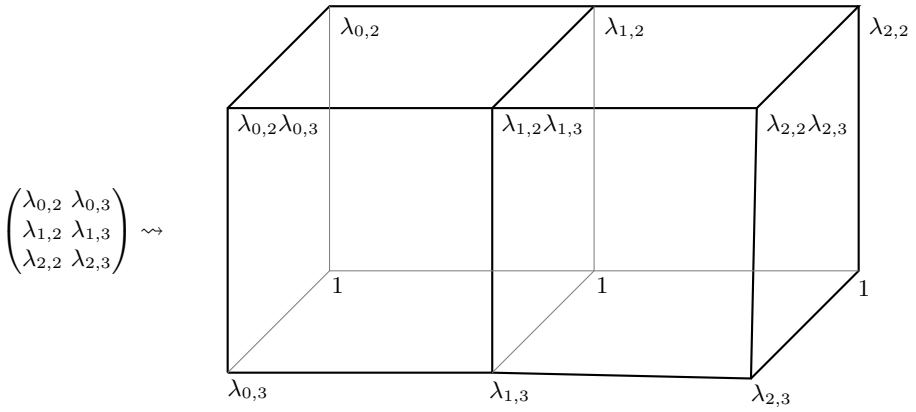

Fig. 1. Two 3×1 vectors defining a $3 \times 2 \times 2$ Vandermonde-Weyman-Zelevinsky tensor.

defining 3×1 vectors as columns of a matrix. Observe that each entry in the tensor is a product of entries of the defining matrix. More generally, for a desired r-dimensional boundary format with length $k_1 + 1$ in the first dimension, one takes a $(k_1 + 1) \times (r-1)$ matrix Λ and it defines a Vandermonde-Weyman-Zelevinsky tensor $\phi\langle\Lambda\rangle$. The entries of $\phi\langle\Lambda\rangle$ are products of powers of elements in Λ (see definition 3). A renowned fact about square Vandermonde matrices is that if and only if the defining vector has distinct entries, the Vandermonde matrix it defines is non-singular. Likewise, Weyman and Zelevinsky showed that $\phi\langle\Lambda\rangle$ is non-singular (see definition 4) if and only if each column of Λ has distinct entries [37, Prop. 7.3]. It does not matter how entries across columns compare.

1.4 Trapdoor generation

The trapdoor generator is given a finite field $\mathbb{F}_q$ and an r-dimensional boundary format $(k_1 + 1) \times (k_2 + 1) \times \ldots \times (k_r + 1)$ that are large enough to meet the

security requirements and does the following. It draws a uniform matrix Λ over the finite field (whose columns have distinct entries) to get a non-singular $\phi\langle\Lambda\rangle$ of the prescribed boundary format. It then draws a r-tuple $X = (X_1, X_2, \ldots, X_r)$ of invertible square matrices, where the first matrix X_1 is constrained to be diagonal. This r-tuple of matrices act on the tensor $\phi\langle\Lambda\rangle$ by multiplication in the respective dimensions (the matrices are drawn of the appropriate size, to do so) resulting in a tensor $\phi\langle\Lambda\rangle^X$ that is isomorphic to $\phi\langle\Lambda\rangle$. The trapdoor $\mathfrak{t} = (\Lambda, X^{-1})$ consists[1] of Λ and the basis change X^{-1}. The twisted tensor $\phi\langle\Lambda\rangle^X$ is the public description of the one-way function. The public tensor $\phi\langle\Lambda\rangle^X$ is output by explicitly writing down its entries (with no reference to Λ or X).

1.5 The one-way function

The public tensor $\phi\langle\Lambda\rangle^X$ defines a multilinear form mapping an r-tuple of vectors to a field element. But if we evaluate this multilinear form at only an $r-1$ tuple of vectors (in two through r dimensions), the evaluation results in a vector (in the first dimension). We will call this multilinear map that evaluates in all but the first dimension as tensor evaluation. The one way function is built from this tensor evaluation, except the input vectors and the output vector are projective vectors. This projectivisation is to exclude trivial collisions that occur due to multilinearity and are also the reason for the "$+1$" terms in our tensor format descriptions. We will denote the d-dimensional projective space as $\mathbb{P}^d := \mathbb{P}(\mathbb{F}_q^{d+1})$. The range of the one way function is the set of projective vectors in first dimension, but restricted to projective coordinates of Hamming weight exactly (k_2+1). We denote this as the Hamming sphere $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$ of radius k_2+1 in the k_1-dimensional projective space $\mathbb{P}^{k_1}$. The domain $\mathcal{D}$ is defined to be the subset of $(r-1)$-tuples of projective vectors $\mathbb{P}^{k_2} \times \mathbb{P}^{k_3} \times \ldots \times \mathbb{P}^{k_r}$ whose image under tensor evaluation lies on the sphere $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$. In summary, the one way function

$$\mathfrak{h}_{\phi\langle\Lambda\rangle^X} : \mathcal{D} \longrightarrow \mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$$

is tensor evaluation restricted to map to the Hamming sphere of radius k_2+1. It is not obvious how to draw samples from this restricted domain $\mathcal{D}$ (to ensure sparse images): looking ahead, this task will be accomplished by our domain samplers. Algorithmically, the domain samplers perform simple linear algebra, yet the proof of correctness (of termination, the uniformity of the image, etc.) is delicate and exploits the structure of the Vandermonde-Weyman-Zelevinsky tensor Λ (which is not visible to the domain samplers).

Our one-way function is in the spirit of, but does not exactly conform to the GPV preimage sampleable trapdoor function framework. Notably, the domain $\mathcal{D}$ depends on the randomness of the trapdoor (even when the security parameter is fixed), whereas the domain in GPV is fixed. Therefore, we do not directly

[1] The trapdoor has the tuple of inverses $X^{-1} = (X_1^{-1}, X_2^{-1}, \ldots, X_r^{-1})$ instead of X for efficiency, so the signing algorithms do not have to perform matrix inversions.

invoke the security proof in GPV, but instead follow a similar path to tailor a security proof to our context.

Writing down the one way function in three dimensions. Let us pause to digest the construction so far, by explicitly writing down the one way function, focusing on $(2k+1) \times (k+1) \times (k+1)$ formats. The trapdoor is a $(2k+1) \times 2$ matrix Λ and a triple of invertible matrices $(X_1, X_2, X_3) \in GL_{2k+1}(\mathbb{F}_q) \times GL_{k+1}(\mathbb{F}_q) \times GL_{k+1}(\mathbb{F}_q)$ with X_1 being diagonal. The public tensor $\phi\langle\Lambda\rangle$ is of $(2k+1) \times (k+1) \times (k+1)$ format. The range of the one way function is the Hamming sphere $\mathcal{S}_{k+1}(\mathbb{P}^{2k})$, consisting of projective vectors of weight just over half the length. Therefore, the size of the range $\binom{2k+1}{k+1}(q-1)^{k+1}$ (surface area of the sphere) is exponential in k. The domain $\mathcal{D}$ consists of pairs of projective vectors in $\mathbb{P}^k \times \mathbb{P}^k$ that map to $\mathcal{S}_{k+1}(\mathbb{P}^{2k})$. Writing the entries of the public tensor as

$$\left(\phi\langle\Lambda\rangle^X_{i_1,i_2,i_3}\right)_{0\leq i_1\leq 2k, 0\leq i_2\leq k, 0\leq i_3\leq k}$$

and a pair of projective vectors in the second and third dimension as $(\widehat{w}^{(2)}, \widehat{w}^{(3)}) \in \mathbb{P}^k \times \mathbb{P}^k$, the one way function reads explicitly as

$$\mathfrak{h}_{\phi\langle\Lambda\rangle^X} : \mathcal{D} \longrightarrow \mathcal{S}_{k+1}(\mathbb{P}^{2k}) \tag{1.1}$$

$$(\widehat{w}^{(2)}, \widehat{w}^{(3)}) \longmapsto \left(\sum_{i_2,i_3} \phi\langle\Lambda\rangle^X_{i_1,i_2,i_3} \widehat{w}^{(2)}_{i_2} \widehat{w}^{(3)}_{i_3}\right)_{0\leq i_1\leq 2k} . \tag{1.2}$$

Given $(\widehat{w}^{(2)}, \widehat{w}^{(3)}) \in \mathcal{D}$, it is easy to compute $\mathfrak{h}_{\phi\langle\Lambda\rangle^X}(\widehat{w}^{(2)}, \widehat{w}^{(3)})$ from merely the knowledge of the public tensor $\phi\langle\Lambda\rangle^X$. Looking ahead, we will devise a domain sampler to sample from $\mathcal{D}$, guaranteeing that the image under $\mathfrak{h}_{\phi\langle\Lambda\rangle^X}$ is uniform.

1.6 Lagrange interpolation in higher dimensions

We extend Lagrange interpolation to higher dimensions in lemma 1, by devising an interpolation algorithm that given a matrix Λ defining a non-singular Vandermonde-Weyman-Zelevinsky tensor $\phi\langle\Lambda\rangle$ and a sparse enough target, finds an $r-1$ tuple of vectors that hit the target under tensor evaluation. By sparse enough, we mean that the Hamming weight is at most the length of the second longest dimension. In particular, it works for targets in the Hamming sphere $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$ which is the range of our one way function. It also works for the Hamming ball $\mathcal{B}_{k_2+1}(\mathbb{P}^{k_1})$ which includes the interior of the sphere. The crucial property we exploit in our interpolation is that tensor evaluation with respect to a Vandermonde-Weyman-Zelevinsky tensor decouples into a product of polynomials, if we think of the coordinates of the input vectors as coefficients of polynomials. This idea is highlighted at the end of subsection 2.2. This allows us to zero out many of the output coordinates of tensor evaluation, by choosing input vectors whose associated polynomials vanish at the corresponding entries of the matrix Λ. After zeroing out enough of the coordinates, the remaining

constraints imposed by the target is satisfied by the usual one dimensional Lagrange interpolation. The need for sparsity in our generalisation of Lagrange interpolation to higher dimensions is curious and warrants further investigation.

1.7 Preimage samplers

We describe a preimage sampler $\mathsf{SamplePre}(\mathfrak{t}, \widehat{w}^{(1)})$ which takes as input the trapdoor $\mathfrak{t} = (\Lambda, X)$ and a target $\widehat{w}^{(1)}$ and computes a preimage of the target under the one way function. The preimage sampler inverts the base change X in the trapdoor, to translate the problem from being phrased in terms of the public tensor $\phi\langle\Lambda\rangle^X$ back to $\phi\langle\Lambda\rangle$. In doing so, it is crucial that X_1 is diagonal, for it preserves the Hamming weight of the target in the translation. Once phrased in terms of $\phi\langle\Lambda\rangle$, the preimage sampler does a randomised version of the aforementioned interpolation. The randomisation is essentially in the choice of which set of zero coordinates are satisfied by which input vectors. There is also a deterministic version, which is useful for instance in encryption, where these choices are made deterministically to always return the same preimage for a given target. Curiously, for three dimensional $(2k+1)\times(k+1)\times(k+1)$ formats, even the randomised preimage sampler turns out to be deterministic. The size of the support preimages are drawn from blows up exponentially for four or more dimensions. Even in three dimensions, a bit of randomness can be injected by permuting the second and third dimensions for $(2k+1)\times(k+1)\times(k+1)$ formats. We present such a variant preimage sampler $\mathsf{SamplePre}^{\mathrm{3DB}}(\mathfrak{t}, \widehat{w}^{(1)})$, which flips a coin to randomly permute the second and third dimensions, before applying $\mathsf{SamplePre}(\mathfrak{t}, \widehat{w}^{(1)})$. Consequently, for each target, $\mathsf{SamplePre}^{\mathrm{3DB}}(\mathfrak{t}, \widehat{w}^{(1)})$ draws uniformly from two distinct preimages.

1.8 Domain samplers

Recall that the domain sampler is given a tensor $\phi\langle\Lambda\rangle^X$ and has to find an $r-1$ tuple of projective vectors that map under tensor evaluation with respect to $\phi\langle\Lambda\rangle^X$ to a sparse vector lying on the Hamming sphere $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$. Since tuples of projective vectors that map to such sparse vectors are exponentially rare, naive rejection sampling fails and we have to resort to some algebra. Further, the domain sampler does not have access to the trapdoor and thus has to treat the input tensor $\phi\langle\Lambda\rangle^X$ as though it were a random tensor. These difficulties make our domain samplers highly non-trivial. We present three domain samplers:

- $\mathsf{SampleDom}^{\mathrm{DB}}(\phi\langle\Lambda\rangle^X)$ for doubly boundary formats in arbitrary dimension,
- $\mathsf{SampleDom}^{\mathrm{3D}}(\phi\langle\Lambda\rangle^X)$ for three dimensional boundary formats,
- $\mathsf{SampleDom}^{\mathrm{3DB}}(\phi\langle\Lambda\rangle^X)$ for three dimensional doubly boundary formats.

Unlike preimage samplers, we do not know how to construct domain samplers in arbitrary boundary formats of four or more dimensions. The common strategy of all three samplers is to pick the support of the image at random and try to zero out the image outside the support. The difference lies in later step. The first algorithm $\mathsf{SampleDom}^{\mathrm{DB}}(\phi\langle\Lambda\rangle^X)$ draws the domain vectors from (3-to-r)-dimensions

at random and solves a linear system to determine the second domain vector. There is subtle rejection sampling to make sure that the algorithm terminates and induces a uniform image. Curiously, the aforementioned decoupling of the Vandermonde-Weyman-Zelevinsky tensors is critical to the proof that the images induced are uniform, although the Vandermonde-Weyman-Zelevinsky tensor structure is not visible to the domain sampling algorithms themselves. The second algorithm $\mathsf{SampleDom}^{\mathrm{3D}}(\phi\langle\Lambda\rangle^X)$ does something similar, except it draws the second domain vector at random and solves for the third. This is only possible in three dimensions, for in higher dimensions a similar strategy would need to solve multilinear systems. The third algorithm $\mathsf{SampleDom}^{\mathrm{3DB}}(\phi\langle\Lambda\rangle^X)$ flips a coin and chooses one of the first two to apply. It only applies for three dimensional doubly boundary formats $(2k+1)\times(k+1)\times(k+1)$. Remarkably, conditioned on a target $\widehat{w}^{(1)}$, the domain sample distribution induced by $\mathsf{SampleDom}^{\mathrm{3DB}}(\phi\langle\Lambda\rangle^X)$ is exactly the same as the preimage distribution induced by the preimage sampler $\mathsf{SamplePre}^{\mathrm{3DB}}(\mathsf{t}, \widehat{w}^{(1)})$. Looking ahead, this equality will help prove that our signatures are unforgeable under the hardness assumptions.

Distinction between preimage and domain samplers. There is another difference between our preimage/domain samplers to appreciate, besides the knowledge/ignorance of the trapdoor. The interpolation algorithm inside the preimage sampler can hit any target on (or in) the Hamming sphere $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$. In contrast, the algorithmic mechanism inside our domain samplers can zero out the coordinates outside the support of any desired target, but in doing so exhaust all the available degrees of freedom, resulting in the non zero coordinates of the image being uniformly random. In Hash-and-Sign signatures, only the preimage sampler is used as part of the algorithms. The domain sampler only appears in the security proof, helping to simulate signatures in the existential unforgeability analysis. In encryption schemes, both the the preimage and the domain sampler play an algorithmic role.

1.9 Hardness assumptions

We make two hardness assumptions which together imply collision resistance of the one-way functions. The first is a pseudorandomness assumption: that a random tensor in the isomorphism class of a random non-degenerate Vandermonde-Weyman-Zelevinsky tensors is computationally indistinguishable from a truly random tensor. This is closely related to the well-studied analogous cryptographic assumptions in tensor isomorphism based cryptosystems [7]. Ours seems like a safer assumption, since we only publish one of the tensors. Whereas tensor isomorphism publishes two tensors and says you still can not tell if it is a random pair of isomorphic tensors or a random pair of tensors. But then, our tensors have more algebraic structure, so the two assumptions are incomparable. The second assumption is that a random multilinear system of equations with a random sparse vector as target is cryptographically hard. We benchmark this hardness by looking to the best known Gröbner basis and high-dimensional resultant based algorithms tailored to solving such multilinear systems [11,33,34].

Applying these benchmarks to $(2k+1)\times(k+1)\times(k+1)$ formats over a finite field with at least $4k$ elements, setting $k \geq 2\lambda$ suffices to meet a security parameter λ. This is a conservative choice, that also takes into account the recent **TI** algorithm (Hull attack) of Couvreir and Levrat [10] applicable to boundary formats (earlier works [26] only apply to cubical formats). For $(2k+1)\times(k+1)\times(k+1)$ boundary formats, their run time is roughly $q^{k/2}$. More generally, their complexity estimate is roughly $q^{k/2}$, with $k+1$ being the length of the shortest dimension. The choice is conservative since it is not clear if a fast **TI** algorithm translates to an algorithm breaking the pseudorandomness assumption, as there are exponentially many Vandermonde-Weyman-Zelevinsky tensor orbits to draw from. As a clarification, our tensors are non singular and do not have the rank deficiency to be vulnerable to the min-rank attacks on the trapdoor from tensors proposal by Schulman over a decade back [32]. Further, our tensor formats and one-way functions are different.

1.10 Hash-and-Sign signatures and encryption

Hash-and-Signatures are one of the most direct and well understood design strategies for digital signatures [6]. We design a simple Hash-and-Sign signature scheme based on the preimage sampleable trapdoor one way functions from tensors. For its instantiation with three dimensional $(2k+1)\times(k+1)\times(k+1)$ formats, we rigorously prove existential unforgeability under chosen-message attacks (EUF-CMA) in the random oracle model (ROM) under our two hardness assumptions. The proof also goes through in the quantum-accessible random oracle model (QROM), accounting for quantum adversaries. Remarkably, the signature size scales as $\mathcal{O}(\lambda\log(\lambda))$ with respect to the security parameter λ. This nearly linearly scaling is close to that of the best lattice and code based signature schemes, such as Dilithium, Falcon and HQC, all of which have been adopted as part of NIST's post-quantum signature standards. The reason signatures are so short is that the preimages are just vectors, and not tensors. Tensor isomorphism based signature schemes built on the Goldreich-Micali-Wigderson protocol, such as ALTEQ and MEDS have to write down a tensor as part of the signature, meaning their signature lengths are cubic in the security parameter. Our public verification key however is a tensor, which takes cubic length in the security parameter λ to write down. The size of the secret signing key is dominated by size of the tuple X of base change matrices, which is quadratic in the security parameter. We could reduce it to linear in the security parameter by replacing the invertible matrices in the basis change with pseudorandom invertible matrix with a linear length description. We finally present a simple CPA-secure encryption scheme, which under the Fujisaki-Okomoto transformation should give a $\mathsf{IND}-\mathsf{CCA}_2$ secure scheme in the random oracle model. We defer the formal analysis of the encryption scheme to a later version of the paper.

1.11 Organisation

In section 2, we establish notation, recall the notion of Vandermonde-Weyman-Zelevinsky tensors and describe our higher dimensional generalisation of Largrange interpolation. This section can be read independent of the other sections, since we anticipate the interpolation algorithm to be of use beyond cryptographic applications. In section 3, we describe the trapdoor generation, preimage sampler and domain sampler algorithms constituting our preimage sampleable trapdoor one way function construction. We conclude the section with the hardness assumptions. In section 5, we present and rigorously analyse our Hash-and-Sign signature scheme and end with a sketch of the encryption scheme.

2 Vandermonde-Weyman-Zelevinsky tensor interpolation

2.1 Tensor notation

Let $\mathbb{F}_q$ denote the finite field with q elements. For a positive integer k, let $\left(\mathbb{F}_q^k\right)^\vee$ denote the dual vector space of $\mathbb{F}_q^k$, consisting of $\mathbb{F}_q$-linear maps from $\mathbb{F}_q^k$ to $\mathbb{F}_q$. To us, an r- dimensional tensor over $\mathbb{F}_q$ of format $(k_1+1)\times(k_2+1)\times\ldots\times(k_r+1)$ (for positive integers $k_1, k_2, \ldots, k_r$) is an element

$$\phi \in \left(\mathbb{F}_q^{k_1+1}\right)^\vee \otimes \left(\mathbb{F}_q^{k_2+1}\right)^\vee \otimes \ldots \otimes \left(\mathbb{F}_q^{k_r+1}\right)^\vee .$$

We will use j exclusively to index dimensions in $\{1, 2, \ldots, r\}$. Without loss of generality, assume $k_1 \geq k_2 \geq \ldots \geq k_r$. Fix an ordered basis for the dual vector spaces, or equivalently, fix a coordinate system $w^{(j)} = \left(w_0^{(j)}, w_1^{(j)}, \ldots, w_{k_j}^{(j)}\right)$ for the j-th vector space $\mathbb{F}_q^{k_j+1}$. With bases fixed, the tensor ϕ is described by an r-dimensional matrix of $\mathbb{F}_q$ elements

$$(\phi_{i_1,i_2,\ldots,i_r})_{0\leq i_1\leq k_1, 0\leq i_2\leq k_2,\ldots,0\leq i_r\leq k_r},$$

which will be the presentation of tensors as inputs or outputs to our algorithms. Think of the tensor ϕ as a multilinear map in two different ways. First, as the multilinear form

$$\begin{aligned} f_\phi : \mathbb{F}_q^{k_1+1} \times \mathbb{F}_q^{k_2+1} \times \ldots \times \mathbb{F}_q^{k_r+1} &\longrightarrow \mathbb{F}_q \\ \left(w^{(1)}, w^{(2)}, \ldots, w^{(r)}\right) &\longmapsto \sum_{\substack{0\leq i_1\leq k_1 \\ \vdots \\ 0\leq i_r\leq k_r}} \phi_{i_1,i_2,\ldots,i_r} w_{i_1}^{(1)} w_{i_2}^{(2)} \ldots w_{i_r}^{(r)}. \end{aligned}$$

associated with evaluating the dual vectors. Second, as the multilinear map

$$\begin{aligned} f_\phi^1 : \mathbb{F}_q^{k_2+1} \times \mathbb{F}_q^{k_3+1} \times \ldots \times \mathbb{F}_q^{k_r+1} &\longrightarrow \mathbb{F}_q^{k_1+1} \\ \left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) &\longmapsto \left(\sum_{\substack{0\leq i_2\leq k_2 \\ \vdots \\ 0\leq i_r\leq k_r}} \phi_{i_1,i_2,\ldots,i_r} w_{i_2}^{(2)} w_{i_3}^{(3)} \ldots w_{i_r}^{(r)} \right)_{0\leq i_1\leq k_1} \end{aligned}$$

that evaluates in all but the first dimension. Denote the i_1-th coordinate of the image as

$$f_\phi^1\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)_{i_1} := \sum_{\substack{0 \leq i_2 \leq k_2 \\ \vdots \\ 0 \leq i_r \leq k_r}} \phi_{i_1, i_2, \ldots, i_r} w_{i_2}^{(2)} w_{i_3}^{(3)} \ldots w_{i_r}^{(r)}.$$

It is this second multilinear map that we will soon call tensor evaluation, but applied to projective vectors.

Definition 1 (Tensor evaluation). *Let $\mathbb{P}^{k_j} := \mathbb{P}(\mathbb{F}_q^{k_j+1})$ denote the projectivisation of $\mathbb{F}_q^{k_j+1}$ and fix the projective coordinate system*

$$\widehat{w}^{(j)} := \left(w_0^{(j)} : w_1^{(j)} : \ldots : w_{k_j}^{(j)}\right) \in \mathbb{P}^{k_j}$$

corresponding to non zero coordinate vectors $w^{(j)}$. Since the multilinear map f_ϕ^1 is given by homogeneous polynomials in the coordinates, the evaluation map

$$\begin{aligned}\hat{f}_\phi^1 : \mathbb{P}^{k_2} \times \mathbb{P}^{k_3} \times \ldots \times \mathbb{P}^{k_r} &\longrightarrow \mathbb{P}^{k_1} \cup \{\mathbf{0}\} \\ \left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) &\longmapsto \left(f_\phi^1\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)_0 : f_\phi^1\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)_1\right. \\ &\qquad \left. : \ldots : f_\phi^1\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)_{k_1}\right).\end{aligned}$$

from projective coordinates to projective coordinates is a well defined map. Except, it can map to the all zero vector, which is not allowed in projective coordinates. We add the zero vector $\mathbf{0}$ to the co-domain to account for this possibility.

While f_ϕ^1 and $\hat{f}_\phi^1$ are in spirit the same map, $\hat{f}_\phi^1$ has a hat in the superscript to emphasise that it is a map on projective coordinates. We will build our trapdoor one way functions with projective coordinates as inputs and outputs, since that resolves the issue of trivial collisions that arise due to multilinearity. Mildly abusing notation, we occasionally also call f_ϕ^1 as tensor evaluation, when it is clear we are dealing with affine coordinate vectors.

Definition 2 (Boundary format, Doubly boundary format). *A tensor format $(k_1+1) \times (k_2+1) \times \ldots \times (k_r+1)$ is called as boundary, if and only if $k_1 = k_2 + k_3 + \ldots + k_r$. A doubly boundary format is a boundary format satisfying the further constraint that $k_2 = k_3 + k_4 + \ldots + k_r$.*

Boundary formats are those that generalise square matrices to higher dimensions in the strictest sense, according to the theory of hyperdeterminants developed by Gelfand, Kapranov and Zelevinsky [15,16]. Boundary formats are the setting for all our constructions. A doubly boundary format is a name we made up, and refers to boundary formats that remain boundary formats even after removing the first dimension.

2.2 Vandermonde-Weyman-Zelevinsky tensors

A Vandermonde matrix is a square two dimensional matrix described by one vector. Analogously, Weyman and Zelevinsky defined r-dimensional boundary format tensors that are described by $r-1$ vectors. These tensors generalise Vandermonde matrices and satisfy similar properties in higher dimensions too. For instance, the tensor is non singular if and only if the describing $r-1$ vectors are each made of distinct coordinates. We will call these as Vandermonde-Weyman-Zelevinsky tensors, and state their definition below. It is convenient to package the $r-1$ vectors into a matrix in the definition.

Definition 3. *(**Vandermonde-Weyman-Zelevinsky tensor.**) Let* $(k_1+1)\times(k_2+1)\times\ldots\times(k_r+1)$ *be an* r*-dimensional boundary format with* $k_1 = k_2 + k_3 + \ldots + k_r$ *and* $r \geq 2$*. The Vandermonde-Weyman-Zelevinsky tensor* $\phi\langle\Lambda\rangle$ *associated with a* $(k_1+1)\times(r-1)$ *matrix*

$$\Lambda = (\lambda_{i_1,j})_{0\leq i_1\leq k_1, 2\leq j\leq r},$$

over a finite field $\mathbb{F}_q$ *is defined as the one with entries*

$$(\phi\langle\Lambda\rangle_{i_1,i_2,\ldots,i_r}) := \left(\lambda_{i_1,2}^{i_2}\lambda_{i_1,3}^{i_3}\ldots\lambda_{i_1,r}^{i_r}\right)_{0\leq i_1\leq k_1, 0\leq i_2\leq k_2,\ldots,0\leq i_r\leq k_r}.$$

The definition also holds in any field, despite only being stated over finite fields. Observe that $\phi\langle\Lambda\rangle_{i_1,i_2,\ldots,i_r}$ is a product of powers of $r-1$ entries of the matrix Λ, and not r entries, since the dimension index j in Λ only ranges from 2 to r. In fact, the entries that appear are precisely from the i_1-indexed row of Λ. Therefore, $\phi\langle\Lambda\rangle_{i_1,i_2,\ldots,i_r}$ is a product of powers of the i_1-indexed row entries of Λ, with the powers determined by the indices $i_2, i_3, \ldots, i_r$. This structure will soon become familiar, when we illustrate the first examples. Specialising to $r=2$ yields the familiar Vandermonde square matrices. The simplest three dimensional examples, $3\times2\times2$ format Vandermonde-Weyman-Zelevinsky tensors are pictured in figure 1.3. The simplest four dimensional boundary format is $4\times2\times2\times2$. But, let us look to the bigger $5\times3\times2\times2$ boundary format in figure 2.2, since the latter is doubly boundary. Since it is difficult to illustrate four dimensional tensors on paper, we draw its five slices in the first dimension. Each slice is a boundary format tensor, since by design the tensor they are sliced from is doubly boundary format. But the slices are not necessarily Vandermonde-Weyman-Zelevinsky tensors.

Decoupling of Vandermonde-Weyman-Zelevinsky tensor evaluations The principal observation that we will exploit repeatedly in our construction is that the tensor evaluation

$$\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) \longmapsto \left(\sum_{\substack{0\leq i_2\leq k_2\\ \ldots\\ 0\leq i_r\leq k_r}} \lambda_{i_1,2}^{i_2}\lambda_{i_1,3}^{i_3}\ldots\lambda_{i_1,r}^{i_r} w_{i_2}^{(2)} w_{i_3}^{(3)}\ldots w_{i_r}^{(r)}\right)_{0\leq i_1\leq k_1},$$

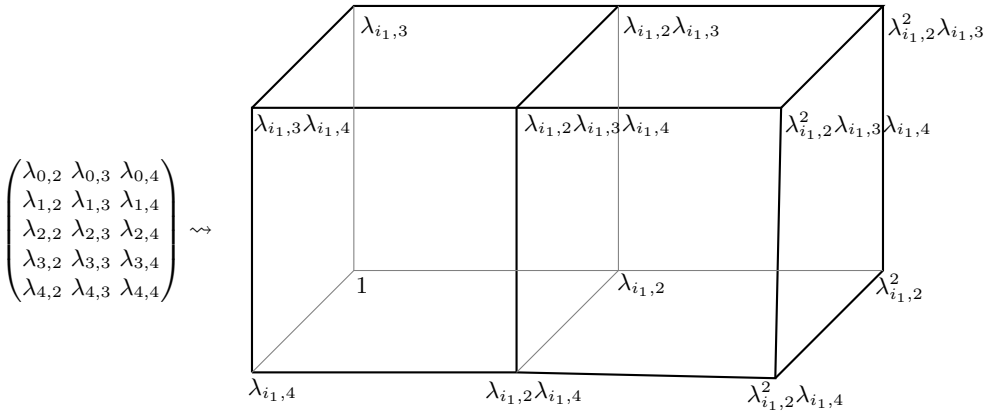

Fig. 2. A 5×3 matrix and the i_1-th slice (in the first dimension) of the $5 \times 3 \times 2 \times 2$ Vandermonde-Weyman-Zelevinsky tensor it defines. There are five such slices, enumerated by $0 \leq i_1 \leq 4$.

with respect to a Vandermonde-Weyman-Zelevinsky tensor $\phi\langle \Lambda \rangle$ splits into a product of polynomials

$$\left(w^{(2)}, w^{(3)}, \ \ldots, w^{(r)}\right) \longmapsto \tag{2.1}$$
$$\left(\left(\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \lambda_{i_1,2}^{i_2}\right)\left(\sum_{i_3=0}^{k_3} w_{i_3}^{(3)} \lambda_{i_1,3}^{i_3}\right) \ldots \left(\sum_{i_r=0}^{k_r} w_{i_r}^{(r)} \lambda_{i_1,r}^{i_r}\right)\right)_{0 \leq i_1 \leq k_1},$$

when we think of the coordinates of the input vectors as polynomials. To clarify, if we think of the j-th dimension input vector $w^{(j)}$ as the polynomial

$$P_j(\Lambda_j) := \sum_{i_j=0}^{k_j} w_{i_j}^{(j)} \Lambda_j^{i_j} \in \mathbb{F}_q[\Lambda_j]$$

in some indeterminate Λ_j, then the tensor evaluation map is the map

$$\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) \longmapsto (P_2(\lambda_{i_1,2}) P_3(\lambda_{i_1,3}) \ldots P_r(\lambda_{i_1,r}))_{0 \leq i_1 \leq k_1} \tag{2.2}$$

evaluating the decoupled multivariate polynomial $P_2(\Lambda_2)P_3(\Lambda_3)\ldots P_r(\Lambda_r)$ at the first row of Λ. This observation was critical in Weyman and Zelevinsky's characterisation of the non-degeneracy of tensors that now bear their name [37, Prop. 7.3], which we are happy to adopt in designing cryptographic primitives.

Degeneracy/Singularity of boundary format tensors. The notion of singularity or degeneracy of matrices have been extended to higher dimensional

tensors [16,15]. While we do not require these notions to describe our algorithms, they help understand the origin of some of the ideas and rule out degeneracy attacks [31,23]. Singularity is defined analytically while degeneracy is defined algebraically. But they are equivalent and we will use the terms interchangeably. Informally, a tensor is defined to be degenerate if there is a tuple of non zero coordinate vectors such that evaluating the tensor at all but one of the vectors gives the all zero dual vector. For boundary formats, due to the Cayley trick, this definition simplifies to only needing to leave out the longest dimension [16,15].

Definition 4 (Singularity/Degeneracy). *A $(k_1+1)\times(k_2+1)\times\ldots\times(k_r+1)$ boundary format tensor is called degenerate or singular if and only if the all zero vector $\mathbf{0}$ is in the image $\hat{f}^1_\phi\left(\mathbb{P}^{k_2}\times\mathbb{P}^{k_3}\times\ldots\times\mathbb{P}^{k_r}\right)$ of the tensor evaluation map.*

Vandermonde matrices are non-singular if and only if the vector definig the matrix is made up of distinct entries. Weyman and Zelevinsky proved the following theorem, with a similar characterisation of singularity in higher dimensions, exploiting the aforementioned decoupling.

Theorem 1 (Singularity of Vandermonde-Weyman-Zelevinsky tensors). *Let $(k_1+1)\times(k_2+1)\times\ldots\times(k_r+1)$ be an r-dimensional boundary format with $k_1 = k_2+k_3+\ldots+k_r$ and $r\geq 2$. The Vandermonde-Weyman-Zelevinsky tensor $\phi\langle\Lambda\rangle$ of such a format associated with a $(k_1+1)\times(r-1)$ matrix $\Lambda=(\lambda_{i_1,j})_{0\leq i_1\leq k_1, 2\leq j\leq r}$, is singular/degenerate if and only if there exists a dimension $j\in\{2,3,\ldots,r\}$ and two distinct indices $i_1,i_1'\in\{0,1,\ldots,k_1\}$ such that $\lambda_{i_1,j}=\lambda_{i_1',j}$.*

Proof. See Weyman and Zelevinsky [37, Prop. 7.3]. The proof in [37] is stated over the field of complex numbers, but their argument also works over finite fields. The only special property of fields they use is that a non zero polynomial in one variable has at most as many zeros as the degree.

2.3 Sparse vector interpolation

Let $(k_1+1)\times(k_2+1)\times\ldots\times(k_r+1)$ be an r-dimensional boundary format with $k_1=k_2+k_3+\ldots+k_r$ and $r\geq 3$. For a positive integer d, let

$$\mathcal{B}_d\left(\mathbb{F}_q^{k_1+1}\right) := \left\{w^{(1)}\in\mathbb{F}_q^{k_1+1} \mid d_H(w^{(1)})\leq d\right\}$$

and

$$\mathcal{S}_d\left(\mathbb{F}_q^{k_1+1}\right) := \left\{w^{(1)}\in\mathbb{F}_q^{k_1+1} \mid d_H(w^{(1)})= d\right\}$$

denote the Hamming ball and sphere of radius d in $\mathbb{F}_q^{k_1+1}$, where $d_H()$ is the Hamming weight measuring the number of non zero coordinates. Likewise, let

$$\mathcal{B}_d\left(\mathbb{P}^{k_1}\right) := \left\{\widehat{w}^{(1)}\in\mathbb{P}^{k_1} \mid d_H(\widehat{w}^{(1)})\leq d\right\}$$

and

$$\mathcal{S}_d\left(\mathbb{P}^{k_1}\right) := \left\{\widehat{w}^{(1)}\in\mathbb{P}^{k_1} \mid d_H(\widehat{w}^{(1)})= d\right\}$$

denote the Hamming ball and sphere of radius d in $\mathbb{P}^{k_1}$.

Lemma 1. *Consider r-dimensional boundary formats $(k_1+1)\times(k_2+1)\times\ldots\times(k_r+1)$ with $r\geq 3$. Assume without loss of generality that the first dimension is the longest. That is, $k_1=k_2+k_3+\ldots+k_r$. There exists a polynomial time algorithm that given*

- *a $(k_1+1)\times(r-1)$ matrix $\Lambda=(\lambda_{i_1,j})_{0\leq i_1\leq k_1, 2\leq j\leq r}$ over a finite field $\mathbb{F}_q$ with each of its columns having distinct coordinates (that is, for all j, $i_1\neq i_1'$ implies $\lambda_{i_1,j}\neq\lambda_{i_1',j}$),*
- *a target non zero coordinate vector $w^{(1)}\in\mathbb{F}_q^{k_1+1}$ with at most k_2+1 non zero coordinates,*

finds an $(r-1)$-tuple of non zero coordinate vectors

$$\left(w^{(2)},w^{(3)},\ldots,w^{(r)}\right)\in\mathbb{F}_q^{k_2+1}\times\mathbb{F}_q^{k_3+1}\times\ldots\times\mathbb{F}_q^{k_r+1}$$

which hits the target under the multilinear map $f^1_{\phi\langle\Lambda\rangle}$ to the first dimension associated with the Vandermonde-Weyman-Zelevinsky tensor $\phi\langle\Lambda\rangle$. That is,

$$f^1_{\phi\langle\Lambda\rangle}\left(w^{(2)},w^{(3)},\ldots,w^{(r)}\right)=w^{(1)}.$$

Proof. Write the constraint to meet in the statement of the lemma in terms of the $(r-1)$-tuple of non zero coordinate vectors $\left(w^{(2)},w^{(3)},\ldots,w^{(r)}\right)\in\mathbb{F}_q^{k_2+1}\times\mathbb{F}_q^{k_3+1}\times\ldots\times\mathbb{F}_q^{k_r+1}$ explicitly as

$$\sum_{\substack{0\leq i_2\leq k_2\\ \ldots\\ 0\leq i_r\leq k_r}}\phi\langle\Lambda\rangle_{i_1,i_2,\ldots,i_r}w^{(2)}_{i_2}w^{(3)}_{i_3}\ldots w^{(r)}_{i_r}=w^{(1)}_{i_1},\ \forall\ 0\leq i_1\leq k_1.$$

Substitute $\phi\langle\Lambda\rangle_{i_1,i_2,\ldots,i_r}=\lambda^{i_2}_{i_1,1}\lambda^{i_3}_{i_1,2}\ldots\lambda^{i_r}_{i_1,r}$ for the entries of the Vandermonde-Weyman-Zelevinsky tensor, to get

$$\sum_{\substack{0\leq i_2\leq k_2\\ \ldots\\ 0\leq i_r\leq k_r}}\lambda^{i_2}_{i_1,1}\lambda^{i_3}_{i_1,2}\ldots\lambda^{i_r}_{i_1,r}w^{(2)}_{i_2}w^{(3)}_{i_3}\ldots w^{(r)}_{i_r}=w^{(1)}_{i_1},\ \forall\ 0\leq i_1\leq k_1,$$

which decouples into products as

$$\left(\sum_{i_2=0}^{k_2}w^{(2)}_{i_2}\lambda^{i_2}_{i_1,2}\right)\left(\sum_{i_3=0}^{k_3}w^{(3)}_{i_3}\lambda^{i_3}_{i_1,3}\right)\ldots\left(\sum_{i_r=0}^{k_r}w^{(r)}_{i_r}\lambda^{i_r}_{i_1,r}\right)=w^{(1)}_{i_1},\ \forall\ 0\leq i_1\leq k_1. \tag{2.3}$$

For $2\leq j\leq r$, consider the following polynomials

$$P_j(\Lambda_j):=\sum_{i_j=0}^{k_j}w^{(j)}_{i_j}\Lambda_j^{i_j}\in\mathbb{F}_q[\Lambda_j]$$

in indeterminates Λ_j with the coordinates of $w^{(j)}$ as the coefficients. The constraints in equation 2.3 are equivalent to the system of polynomial equations

$$P_2(\lambda_{i_1,2})P_3(\lambda_{i_1,3})\dots P_r(\lambda_{i_1,r}) = w^{(1)}_{i_1},\ \forall 0 \leq i_1 \leq k_1. \tag{2.4}$$

Partition the set of indices in the first dimension

$$I_2 \cup I_3 \cup \ldots \cup I_r = \{0, 1, \ldots, k_1\}$$

into disjoint non empty subsets $I_2, I_3, \ldots, I_r$ such that

$$|I_2| = k_2 + 1, |I_3| = k_3, |I_4| = k_4, \ldots, |I_r| = k_r \ \text{ and } \ w^{(1)}_{i_1} = 0, \forall i_1 \in \bigcup_{j=3}^{r} I_j.$$

Such a partition is always possible by sweeping all the non zero coordinates of $w^{(1)}$ under I_2, since $k_1 + 1 = k_2 + 1 + k_3 + \ldots + k_r$ and $w^{(1)}$ has at most $k_2 + 1$ non zero coordinates. Set

$$P_j(\Lambda_j) \leftarrow \prod_{i_1 \in I_j} (\Lambda_j - \lambda_{i_1,j}), \forall j \in \{3, 4, \ldots, r\},$$

thereby satisfying

$$P_2(\lambda_{i_1,2})P_3(\lambda_{i_1,3})\dots P_r(\lambda_{i_1,r}) = w^{(1)}_{i_1}(=0),\ \forall i_1 \in I_3 \cup I_4 \cup \ldots \cup I_r.$$

irrespective of $P_2(\Lambda_2)$. The coordinates $\left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$ thus determined can be written down by polynomial multiplication. Further, each of the coordinate vectors in the tuple $\left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$ has at least one non zero coordinate, since the subsets $I_3, I_4, \ldots, I_r$ are all non empty. To satisfy equation 2.4, all that is left to do is find $P_2(\Lambda_2)$ such that,

$$P_2(\lambda_{i_1,2})P_3(\lambda_{i_1,3})\dots P_r(\lambda_{i_1,r}) = w^{(1)}_{i_1},\ \forall i_1 \in I_2. \tag{2.5}$$

Since I_2 and $I_3 \cup I_4 \cup \ldots \cup I_r$ are disjoint, and each column of Λ has distinct coordinates,

$$P_3(\lambda_{i_1,3})P_4(\lambda_{i_1,4})\dots P_r(\lambda_{i_1,r}) \neq 0,\ \forall i_1 \in I_2.$$

Further, $(P_3(\lambda_{i_1,3})P_4(\lambda_{i_1,4})\dots P_r(\lambda_{i_1,r}),\ \forall i_1 \in I_2)$ can be computed by polynomial evaluation and then taking products. Therefore, we may phrase the constraint 2.5 as

$$P_2(\lambda_{i_1,2}) = \frac{w^{(1)}_{i_1}}{P_3(\lambda_{i_1,3})P_4(\lambda_{i_1,4})\dots P_r(\lambda_{i_1,r})},\ \forall i_1 \in I_2, \tag{2.6}$$

which is a polynomial interpolation problem in one variable with degree k_2 and $k_2 + 1$ distinct interpolation points. The interpolation problem has a solution, for instance through Lagrange interpolation,

$$P_2(\Lambda_2) \leftarrow \sum_{i_1 \in I_2} \frac{w^{(1)}_{i_1}}{P_3(\lambda_{i_1,3})P_4(\lambda_{i_1,4})\dots P_r(\lambda_{i_1,r})} \prod_{i'_1 \in I_2, i'_1 \neq i_1} \frac{\Lambda_2 - \lambda_{i'_1,2}}{\lambda_{i_1,2} - \lambda_{i'_1,2}}$$

and can be found in nearly linear time using fast Lagrange interpolation [14]. Since the target is a non zero vector, at least one of interpolated values is non zero, implying the interpolated polynomial $P_2(\Lambda_2)$ is non zero. □

3 Preimage sampleable trapdoor one-way functions

We begin by informally recounting the abstraction of the preimage sampleable trapdoor one-way functions proposed in [17], through its three algorithms.

1. **Trapdoor generator** $\mathsf{TrapGen}$**.** $\mathsf{TrapGen}$ is a randomised polynomial time algorithm that takes some parameters (sufficient to meet a desired security parameter 1^λ) as input and generates a one-way function-trapdoor pair

$$(\mathfrak{h} : \mathcal{D} \to \mathcal{R}, \mathfrak{t}),$$

   where both the function $\mathfrak{h}$ and trapdoor $\mathfrak{t}$ have short descriptions (so as to be treated as inputs/outputs of algorithms). Further, the function is easy to compute. The domain $\mathcal{D}$ and range $\mathcal{R}$ are recognisable finite sets, where recognisable means there are polynomial time algorithms that test membership. Concretely, the elements of these sets are encoded as bit strings and there are polynomial time algorithms to decide membership of a given bit string.

2. **Domain sampler** $\mathsf{SampleDom}$**.** The domain sampler $\mathsf{SampleDom}(\mathfrak{h})$ is a randomised polynomial time algorithm parametrised by the function $\mathfrak{h} : \mathcal{D} \to \mathcal{R}$, that draws a sample $x \leftarrow \mathsf{SampleDom}(\mathfrak{h})$ from $\mathcal{D}$ such that the image $\mathfrak{h}(x)$ is uniform in $\mathcal{R}$. The induced sample distribution does not necessarily have to be uniform in $\mathcal{D}$. In [17], they settle for the weaker guarantee of being statistically close to uniform for the lattice based constructions, and the weaker notion suffices for some applications. But we will construct domain samplers whose outputs are exactly uniform, meeting the stronger guarantee.

3. **Trapdoor preimage sampler** $\mathsf{SamplePre}$**.** The trapdoor preimage sampler $\mathsf{SamplePre}(\mathfrak{t}, y)$ is a randomised polynomial time algorithm parametrised by the function-trapdoor pair. Given a target image $y \in \mathcal{R}$ as input, it outputs a preimage $x \leftarrow \mathsf{SamplePre}(\mathfrak{t}, y) \in \mathcal{D}$. That is, $\mathfrak{h}(x) = y$. We want the preimages to be drawn with enough randomness so as to reveal as little about the trapdoor as permitted, when repeatedly called in certain applications such as signing many messages using the same key. We assume that the function description $\mathfrak{h}$ can easily be computed from the trapdoor $\mathfrak{t}$. This is indeed true in our case. Otherwise, the function description can be added as an input to $\mathsf{SamplePre}$.

Preimage and collision resistance. It must be hard to compute a preimage under the function $\mathfrak{h}$, given only the function $\mathfrak{h}$ and the target image, without knowledge of the trapdoor. Additionally, we may seek collision resistance, requiring it must be hard given the function $\mathfrak{h}$ to compute two distinct $x, x' \in \mathcal{D}$ such that $\mathfrak{h}(x) = \mathfrak{h}(x')$.

We next present the construction in great generality accommodating formats that are either doubly boundary formats (in arbitrary dimensions) or three dimensional boundary formats.

3.1 Trapdoor generation

Tuples of invertible matrices (of the right formats) take tensors to other tensors of the same format as follows.

Definition 5. *(**Basis change**) Tuples* $X = (X_1, X_2, \ldots, X_r) \in \prod_{j=1}^{r} GL_{k_j+1}(\mathbb{F}_q)$ *of invertible matrices act on* $(k_1+1) \times (k_2+1) \times \ldots \times (k_r+1)$ *format tensors* ϕ *as*

$$((X_1, X_2, \ldots, X_r), \phi) \longmapsto \phi^{(X_1, X_2, \ldots, X_r)},$$

where $\phi^{(X_1, X_2, \ldots, X_r)}$ *is defined as the tensor associated with the multilinear form*

$$f_{\phi^{(X_1, X_2, \ldots, X_r)}} : \mathbb{F}_q^{k_1+1} \times \mathbb{F}_q^{k_2+1} \times \ldots \times \mathbb{F}_q^{k_r+1} \longrightarrow \mathbb{F}_q$$
$$\left(w^{(1)}, w^{(2)}, \ldots, w^{(r)}\right) \longmapsto f_\phi\left(X_1 w^{(1)}, X_2 w^{(2)}, \ldots, X_r w^{(r)}\right).$$

We will at times denote $\phi^{(X_1, X_2, \ldots, X_r)}$ *as* ϕ^X*, for brevity.*

In essence, the tuple of matrices twist the associated multilinear form by first multiplying the input vector in each dimension by the corresponding matrix. Since

$$\psi = \phi^{(X_1, X_2, \ldots, X_r)} \Leftrightarrow \phi = \psi^{(X_1^{-1}, X_2^{-1}, \ldots, X_r^{-1})},$$

ψ is in the $\prod_{j=1}^{r} GL_{k_j+1}(\mathbb{F}_q)$ orbit of ϕ if an only if ψ is in the $\prod_{j=1}^{r} GL_{k_j+1}(\mathbb{F}_q)$ orbit of ϕ. Therefore the following is well defined.

Definition 6. *(**Tensor isomorphism**) Two tensors of the same format* $(k_1+1) \times (k_2+1) \times \ldots \times (k_r+1)$ *are called isomorphic if and only if they are in the same* $\prod_{j=1}^{r} GL_{k_j+1}(\mathbb{F}_q)$ *orbit.*

From the symmetry of the definition and the transitivity of orbit membership, it is apparent that tensor isomorphism is an equivalence relation.

Let $D_{k_1+1}(\mathbb{F}_q^*) \subset GL_{k_1+1}(\mathbb{F}_q)$ denote the subgroup of diagonal matrices with no zeroes on the diagonal. This subgroup preserves the Hamming weight of the vectors under multiplication. We will use restricted isomorphisms for base change while generating public keys, to preserve Hamming weight in the first dimension.

Definition 7. *(**Restricted tensor isomorphism**) Two tensors of the same format* $(k_1+1) \times (k_2+1) \times \ldots \times (k_r+1)$ *are called restricted isomorphic if and only if they are in the same* $D_{k_1+1}(\mathbb{F}_q^*) \times \prod_{j=2}^{r} GL_{k_j+1}(\mathbb{F}_q)$ *orbit.*

We may act by a bigger subgroup that preserves Hamming weight in the first dimension, by taking products of diagonal and permutation matrices. But that would be vacuous in our construction, since permutations in the first dimension will be accounted for by permuting the random choice of evaluations points in the Vandermonde-Weyman-Zelevinsky tensor. The slices in the first dimension of the Vandermonde-Weyman-Zelevinsky tensor are rank deficient. The action (diagonal, or otherwise) on the first dimension does not change the rank of those

slices, due to the invariance of rank. Instead, it is the $\prod_{j=2}^{r} GL_{k_j+1}(\mathbb{F}_q)$ action in the other dimensions that scrambles and raises the rank of the slices in the first dimension. These actions are general enough to suggest heuristically that the slices in the first dimension are full rank with high probability.

Algorithm 1: $\mathsf{TrapGen}\left(q, 1^{k_1}, 1^{k_2}, \ldots, 1^{k_r}\right)$
Trapdoor generator

Input: A finite field $\mathbb{F}_q$ and an r-dimensional boundary format $(k_1+1) \times (k_2+1) \times \ldots \times (k_r+1)$ with $k_1 = k_2 + k_3 + \ldots + k_r$ and $r \geq 3$; chosen to meet a security parameter 1^λ requirement.

Output: A function-trapdoor pair $(\mathfrak{h}_\psi : \mathcal{D} \longrightarrow \mathcal{R}, \mathfrak{t})$.

1 Draw a uniform $(k_1+1) \times (r-1)$ matrix $\Lambda = (\lambda_{i_1,j})_{0 \leq i_1 \leq k_1, 2 \leq j \leq r}$ over $\mathbb{F}_q$ such that for each column j, $i_1 \neq i_1'$ implies $\lambda_{i_1,j} \neq \lambda_{i_1',j}$.

2 Draw a uniform r-tuple of invertible matrices

$$X = (X_1, X_2, \ldots, X_r) \in D_{k_1+1}(\mathbb{F}_q^*) \times \prod_{j=2}^{r} GL_{k_j+1}(\mathbb{F}_q).$$

3 Set the trapdoor $\mathfrak{t} \leftarrow \left(\Lambda, X^{-1}\right)$, where $X^{-1} := (X_1^{-1}, X_2^{-1}, \ldots, X_r^{-1})$.

4 Twist the Vandermonde-Weyman-Zelevinsky tensor $\phi\langle\Lambda\rangle$ by the basis change $X = (X_1, X_2, \ldots, X_r)$ to get $\psi \leftarrow \phi\langle\Lambda\rangle^X$, explicitly described by its coordinates

$$(\psi_{i_1,i_2,\ldots,i_r}) := \left(\phi\langle\Lambda\rangle^X_{i_1,i_2,\ldots,i_r}\right)_{0 \leq i_1 \leq k_1, 0 \leq i_2 \leq k_2, \ldots, 0 \leq i_r \leq k_r}.$$

5 Output $(\psi, \mathfrak{t})$.

The ψ in the output is thought of as describing $\mathfrak{h}_\psi : \mathcal{D} \longrightarrow \mathcal{R}$ where

- the range $\mathcal{R}$ is the Hamming sphere $\mathcal{S}_{k_2+1}\left(\mathbb{P}^{k_1}\right)$ of radius k_2+1,
- the domain $\mathcal{D}$ is the fibre

$$\left\{ \left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) \in \mathbb{P}^{k_2} \times \mathbb{P}^{k_3} \times \ldots \times \mathbb{P}^{k_r} \right.$$
$$\left. \mid \hat{f}^1_{\phi(\Lambda)^X}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) \in \mathcal{S}_{k_2+1}\left(\mathbb{P}^{k_1}\right) \right\}$$

  of preimages of the Hamming sphere $\mathcal{S}_{k_2+1}\left(\mathbb{P}^{k_1}\right)$ under $\hat{f}^1_{\phi(\Lambda)^X}$.
- the one-way function $\mathfrak{h}_\psi$ is the restriction

$$\mathfrak{h}_\psi : \mathcal{D} \longrightarrow \mathcal{R}$$
$$\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) \longmapsto \hat{f}^1_{\phi(\Lambda)^X}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)$$

  of the tensor evaluation associated with $\phi\langle\Lambda\rangle^X$ to $\mathcal{D}$. As an explicit expression to aid implementation, given the entries of the tensor $\phi\langle\Lambda\rangle^X$, the

one-way function maps

$$\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) \longmapsto \left(\sum_{\substack{0 \leq i_2 \leq k_2 \\ \ldots \\ 0 \leq i_r \leq k_r}} \phi\langle \Lambda \rangle^X_{i_1, i_2, \ldots, i_r} \widehat{w}^{(2)}_{i_2} \widehat{w}^{(3)}_{i_3} \ldots \widehat{w}^{(r)}_{i_r} \right)_{0 \leq i_1 \leq k_1}$$

We choose to include the tuple of inverses X^{-1} as part of the trapdoor (instead of just the tuple of matrices X) to speed up the trapdoor preimage sampler, since preimages for many targets are sought while signing many signatures. Looking ahead, the trapdoor preimage sampler will only need X^{-1}. Whether to perform these inversions just once during trapdoor generation (as we have done) or (as a preprocessing step) during trapdoor preimage sampling may be different for other applications.

3.2 Trapdoor preimage sampling with randomised interpolation

Definition 8. *(**Preimage, Preimage sampling**) Given an r-dimensional boundary format $(k_1+1) \times (k_2+1) \times \ldots \times (k_r+1)$ tensor ψ with $k_1 = k_2 + k_3 + \ldots + k_r$ and a target $\widehat{w}^{(1)} \in \mathcal{B}_{k_2+1}\left(\mathbb{P}^{k_1}\right)$, sample from the preimages*

$$Pre_\psi\left(\widehat{w}^{(1)}\right) := \left\{ \left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) \in \mathbb{P}^{k_2} \times \mathbb{P}^{k_3} \times \ldots \times \mathbb{P}^{k_r} \;\middle|\; \hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) = \widehat{w}^{(1)} \right\}$$

of the target under tensor evaluation $\hat{f}^1_\psi$.

Theorem 2. *(**Trapdoor preimage sampling**) Consider r-dimensional boundary formats $(k_1+1) \times (k_2+1) \times \ldots \times (k_r+1)$ with $r \geq 3$. Assume without loss of generality that the first dimension is the longest. That is, $k_1 = k_2 + k_3 + \ldots + k_r$. There exists an $\widetilde{O}(k_1^2 \log q)$ time [2].*

- *a $(k_1+1) \times (r-1)$ matrix $\Lambda = (\lambda_{i_1,j})_{0 \leq i_1 \leq k_1, 2 \leq j \leq r}$ over a finite field $\mathbb{F}_q$ with each of its columns having distinct coordinates (that is, for all j, $i_1 \neq i_1'$ implies $\lambda_{i_1,j} \neq \lambda_{i_1',j}$),*
- *an r-tuple $X^{-1} = (X_1^{-1}, X_2^{-1}, \ldots, X_r^{-1}) \in D_{k_1+1}(\mathbb{F}_q^*) \times \prod_{j=2}^r GL_{k_j+1}(\mathbb{F}_q)$ of invertible matrices.*
- *a target $\widehat{w}^{(1)} \in \mathcal{B}_{k_2+1}\left(\mathbb{P}^{k_1}\right)$ of weight $d_H(\widehat{w}^{(1)}) \leq k_2 + 1$,*

samples uniformly from a size $\binom{k_1+1-d_H\left(\widehat{w}^{(1)}\right)}{k_3+k_4+\ldots+k_r}\binom{k_3+k_4+\ldots+k_r}{k_3,k_4,\ldots,k_r}$ subset of the preimages $Pre_{\phi\langle\Lambda\rangle^X}\left(\widehat{w}^{(1)}\right)$ with respect to the public tensor $\phi\langle\Lambda\rangle^X$.

[2] The runtime is in bit complexity where $\widetilde{O}$ suppresses logarithmic factors. The extra $\log q$ term disappears in the $\mathbb{F}_q$ arithmetic circuit model.

Proof. Lift the target $\widehat{w}^{(1)}$ to $w^{(1)} \in \mathbb{F}_q^{k_1+1}$. Tuples of coordinate vectors

$$\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) \in \mathbb{F}_q^{k_2+1} \times \mathbb{F}_q^{k_3+1} \times \ldots \times \mathbb{F}_q^{k_r+1}$$

such that

$$f^1_{\phi\langle\Lambda\rangle}\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) = X_1^{-1} w^{(1)}.$$

are in one to one correspondence with tuples of coordinate vectors

$$\left(X_2^{-1} w^{(2)}, X_3^{-1} w^{(3)}, \ldots, X_r^{-1} w^{(r)}\right)$$

such that

$$f^1_{\phi\langle\Lambda\rangle^X}\left(X_2^{-1} w^{(2)}, X_3^{-1} w^{(3)}, \ldots, X_r^{-1} w^{(r)}\right) = w^{(1)}.$$

Since we know the trapdoor basis change $X^{-1} = (X_1^{-1}, X_2^{-1}, \ldots, X_r^{-1})$, it is easy to translate between these two sides of the correspondence. Therefore, it suffices to solve for $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$ such that $f^1_{\phi\langle\Lambda\rangle}\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) = X_1^{-1} w^{(1)}$, from which we output $\left(X_2^{-1} w^{(2)}, X_3^{-1} w^{(3)}, \ldots, X_r^{-1} w^{(r)}\right)$ (with each of its vectors projectivised) as a valid output that lies in the preimage $\mathrm{Pre}_{\phi\langle\Lambda\rangle^X}\left(\widehat{w}^{(1)}\right)$.

Since we know the trapdoor tensor description Λ and the target $X^{-1} w^{(1)}$, we may invoke Lemma 1 to solve for $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$. But we want more than a preimage, we want to sample uniformly from a large enough subset of the preimages. To this end, we will next present Algorithm 2, a randomised version of the algorithm in Lemma 1.

The proof that the output of Algorithm 2 is indeed in $\mathrm{Pre}_{\phi\langle\Lambda\rangle^X}\left(\widehat{w}^{(1)}\right)$ follows from an appropriate base change and the proof of Lemma 1, mutatis mutandis. But we quickly sketch the main ideas for completeness. Through steps 3 and 4, we determine a tuple $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$ of coordinate vectors encoded as a tuple of polynomials

$$\left(\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \Lambda_2^{i_2}, \sum_{i_3=0}^{k_3} w_{i_3}^{(3)} \Lambda_3^{i_3}, \ldots, \sum_{i_r=0}^{k_r} w_{i_r}^{(r)} \Lambda_r^{i_r}\right)$$

in the indeterminates $\Lambda_2, \Lambda_3, \ldots, \Lambda_r$. As highlighted in equation 2.3, the constraint

$$f^1_{\phi\langle\Lambda\rangle}\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) = \left(X_1^{-1} w^{(1)}\right)_{i_1}, \ \forall i_1 \in \{1, 2, \ldots, k_1\} \tag{3.1}$$

we need to meet decouples into

$$\left(\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \lambda_{i_1,2}^{i_2}\right)\left(\sum_{i_3=0}^{k_3} w_{i_3}^{(3)} \lambda_{i_1,3}^{i_3}\right) \ldots \left(\sum_{i_r=0}^{k_r} w_{i_r}^{(r)} \lambda_{i_1,r}^{i_r}\right) = \left(X_1^{-1} w^{(1)}\right)_{i_1}, \tag{3.2}$$
$$\forall i_1 \in \{1, 2, \ldots, k_1\}.$$

Algorithm 2: $\mathsf{SamplePre}\left(\mathsf{t}, \widehat{w}^{(1)}\right)$
Trapdoor preimage sampler for boundary formats

Input: Trapdoor tensor description $\Lambda = (\lambda_{i_1,j})_{0\le i_1\le k_1, 2\le j\le r}$, trapdoor basis change $X^{-1} = (X_1^{-1}, X_2^{-1}, \ldots, X_r^{-1}) \in D_{k_1+1}(\mathbb{F}_q^*) \times \prod_{j=2}^{r} GL_{k_j+1}(\mathbb{F}_q)$, and a target $\widehat{w}^{(1)} \in \mathcal{B}_{k_2+1}\left(\mathbb{P}^{k_1}\right)$.

Output: A sample from $\mathrm{Pre}_{\phi\langle\Lambda\rangle X}\left(\widehat{w}^{(1)}\right)$.

1 Lift the target $\widehat{w}^{(1)}$ to $w^{(1)} \in \mathbb{F}_q^{k_1+1}$ and let $Z := \{i_1 \mid w_{i_1}^{(1)} = 0\}$ denote the indices corresponding to the zero coordinates of the target.

2 Draw a uniform sequence $(I_3, I_4, \ldots, I_r)$ of disjoint subsets $I_3, I_4 \ldots, I_r \subseteq Z$ such that $|I_3| = k_3, |I_4| = k_4, \ldots, |I_r| = k_r$.

3 For every $j \in \{3, 4, \ldots, r\}$, multiply out the polynomial $\prod_{i_1\in I_j}(\Lambda_j - \lambda_{i_1,j}) \in \mathbb{F}_q[\Lambda_j]$ in the indeterminate Λ_j as

$$\sum_{i_j=0}^{k_j} w_{i_j}^{(j)} \Lambda_j^{i_j} \leftarrow \prod_{i_1\in I_j} (\Lambda_j - \lambda_{i_1,j})$$

and read out the polynomial coefficients as the coordinate vector $w^{(j)}$.

4 Set $I_2 \leftarrow \{1, 2, \ldots, k_1\} \setminus \bigcup_{j=3}^{r} I_j$. Read off the coordinate vector $w^{(2)}$ as the coefficients of the polynomial

$$\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \Lambda_2^{i_2} \leftarrow \sum_{i_1\in I_2} \frac{\left(X_1^{-1} w^{(1)}\right)_{i_1}}{\prod_{j=3}^{r}\prod_{i_j\in I_j}\left(\lambda_{i_1,j} - \lambda_{i_j,j}\right)} \prod_{i_1'\in I_2, i_1'\neq i_1} \frac{\Lambda_2 - \lambda_{i_1',2}}{\lambda_{i_1,2} - \lambda_{i_1',2}},$$

where $\left(X_1^{-1} w^{(1)}\right)_{i_1}$ is the i_1-th coordinate of $X_1^{-1} w^{(1)}$.

5 Output the tuple (after projectivising each of its vectors)

$$\left(X_2^{-1} w^{(2)}, X_3^{-1} w^{(3)}, \ldots, X_r^{-1} w^{(r)}\right).$$

Since X_1 is a diagonal matrix with non zero entries on the diagonal, so is X_1^{-1}. Therefore, the support of $w^{(1)}$ and $X_1^{-1} w^{(1)}$ are exactly the same. The choice of $\left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$ made in Step 3 ensures that,

$$\left(\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \lambda_{i_1,2}^{i_2}\right)\left(\sum_{i_3=0}^{k_3} w_{i_3}^{(3)} \lambda_{i_1,3}^{i_3}\right)\cdots\left(\sum_{i_r=0}^{k_r} w_{i_r}^{(r)} \lambda_{i_1,r}^{i_r}\right) = \left(\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \lambda_{i_1,2}^{i_2}\right)\prod_{j=3}^{r}\prod_{i_j\in I_j}\left(\lambda_{i_1,j} - \lambda_{i_j,j}\right),\ \forall i_1 \in \{0, 1, \ldots, k_1\},$$

which vanishes for $i_1 \in \bigcup_{j=3}^{r} I_r$. Thereby, the constraint 3.1 is satisfied for all $i_1 \in \bigcup_{j=3}^{r} I_r$, leaving the constraint yet to be enforced only for the indices $i_1 \in I_2$. With $\left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$ already determined, Step 4 is Lagrange interpolation

to determine the unique $w^{(2)}$ encoded as $\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \Lambda_2^{i_2}$ that satisfies the constraint 3.1 for $i_1 \in I_2$. Concretely, $\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \Lambda_2^{i_2}$ is the unique degree at most k_2 polynomial

$$\sum_{i_2=0}^{k_2} w_{i_2}^{(2)} \Lambda_2^{i_2} = \sum_{i_1 \in I_2} \frac{\left(X_1^{-1} w^{(1)}\right)_{i_1}}{\prod_{j=3}^{r} \prod_{i_j \in I_j} \left(\lambda_{i_1,j} - \lambda_{i_j,j}\right)} \prod_{i_1' \in I_2, i_1' \neq i_1} \frac{\Lambda_2 - \lambda_{i_1',2}}{\lambda_{i_1,2} - \lambda_{i_1',2}}$$

that interpolates the $k_2 + 1$ points

$$\left(\lambda_{i_1,2}, \frac{\left(X_1^{-1} w^{(1)}\right)_{i_1}}{\prod_{j=3}^{r} \prod_{i_j \in I_j} \left(\lambda_{i_1,j} - \lambda_{i_j,j}\right)} \right)_{i_1 \in I_2}.$$

The constraint on each column of Λ having distinct coordinates ensures that none of the denominators in the Largrange interpolation formula vanish.

All that remains is to justify the sample size and run time claims. Step 2 is the only one inducing randomness into the algorithm. There are

$$\binom{k_1 + 1 - d_H\left(\widehat{w}^{(1)}\right)}{k_3 + k_4 + \ldots + k_r} \binom{k_3 + k_4 + \ldots + k_r}{k_3, k_4, \ldots, k_r}$$

choices in picking the sequence of subsets $(I_3, I_4, \ldots, I_r)$, where the term on the right is the multinomial coefficient. Since a polynomial of degree k_j has at most k_j roots and $I_3, I_4, \ldots, I_r$ are all disjoint, no two choices $(I_3, I_4, \ldots, I_r)$ gives rise to the same $\left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$. Therefore, the map $(I_3, I_4, \ldots, I_r) \longmapsto \left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$ is injective. Further, for each $j \in \{3, 4, \ldots, r\}$, by setting $\sum_{i_j=0}^{k_j} w_{i_j}^{(j)} \Lambda_j^{i_j}$ as a monic polynomial $\prod_{i_1 \in I_j} (\Lambda_j - \lambda_{i_1,j})$, we chose a unique representative $w^{(j)}$ for the projectivisation $\widehat{w}^{(j)}$. Therefore, $(I_3, I_4, \ldots, I_r) \longmapsto \left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)$ is injective and we indeed sample uniformly from a preimage sample space of the claimed size.

Finally, we argue for the claimed run time. In step 3, we have $r-2$ polynomial multiplications of the form $\prod_{i_1 \in I_j} (\Lambda_j - \lambda_{i_1,j})$. Each of these can be computed in $\widetilde{O}(k_j \log q)$ time using Fast Fourier multiplication. In total, these polynomial multiplications take $\widetilde{O}(k_1 \log q)$ time. The bottleneck in step 4 is to compute the denominators in the evaluations

$$\left(\frac{\left(X_1^{-1} w^{(1)}\right)_{i_1}}{\prod_{j=3}^{r} \prod_{i_j \in I_j} \left(\lambda_{i_1,j} - \lambda_{i_j,j}\right)} \right)_{i_1 \in I_2},$$

which takes $O\left(\sum_{j=3}^{r} \left(\left(k_2 k_j + k_j^2 + r k_2\right) \log q\right)\right) = O\left(k_1^2 \log q\right)$ time. Given these evaluations, the fast Lagrange interpolation algorithm finds $x^{(2)}$ in $\widetilde{O}(k_2 \log q)$ time [14]. Computing $\left(X_2^{-1} w^{(2)}, X_3^{-1} w^{(3)}, \ldots, X_r^{-1} w^{(r)}\right)$ from $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$ involves matrix inversion or solving linear systems, leading to the bottleneck runtime $O\left(\sum_{j=2}^{r} k_j^{\omega} \log q\right) = O\left(k_1^{\omega} \log q\right)$. □

The following corollary to theorem 2 says that the preimage sampling problem with respect to a tensor ψ restricted isomorphic to some non singular Vandermonde-Weyman-Zelevinsky tensor always has a solution. Of course, given only the tensor ψ without a trapdoor, the preimages are hard to find.

Corollary 1. *For a tensor ψ that is restricted isomorphic to some non singular Vandermonde-Weyman-Zelevinsky tensor, $\mathcal{B}_{k_2+1}\left(\mathbb{P}^{k_1}\right) \subseteq \hat{f}^1_\phi\left(\mathbb{P}^{k_2} \times \mathbb{P}^{k_3} \times \ldots \times \mathbb{P}^{k_r}\right)$.*

Proof. Say $\psi = \phi\langle\Lambda\rangle^X$ for some non-singular Vandermonde-Weyman-Zelevinsky tensor $\phi\langle\Lambda\rangle$ with trapdoor Λ, X^{-1}. Applying theorem 2 with the trapdoor Λ, X^{-1} and some target in $\mathcal{B}_{k_2+1}\left(\mathbb{P}^{k_1}\right)$ as input, it is clear there exists a solution for every target, due to the success of the algorithm. □

In three dimensions, restricted to targets on the Hamming sphere $\mathcal{S}_{k_2+1}\left(\mathbb{P}^{k_1}\right)$, SamplePre is deterministic, drawing from a unique preimage for each target input. Since we need to draw from a bigger sample for our signature schemes, we present the following twisted preimage sampler $\mathsf{SamplePre}^{\mathrm{3DB}}$, whose outputs have at least one bit of entropy.

Algorithm 3: $\mathsf{SamplePre}^{\mathrm{3DB}}\left(\mathsf{t}, \widehat{w}^{(1)}\right)$
Trapdoor preimage sampler for $(2k+1) \times (k+1) \times (k+1)$ formats

Input: Trapdoor tensor description $\Lambda = (\lambda_{i_1,j})_{0 \le i_1 \le 2k, 2 \le j \le 3}$, trapdoor basis change $X^{-1} = (X_1^{-1}, X_2^{-1}, X_3^{-1}) \in D_{2k+1}(\mathbb{F}_q^*) \times GL_{k+1}(\mathbb{F}_q) \times GL_{k+1}(\mathbb{F}_q)$, and a target $\widehat{w}^{(1)} \in \mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$.

Output: A sample from $\mathrm{Pre}_{\phi\langle\Lambda\rangle^X}\left(\widehat{w}^{(1)}\right)$.

1 Lift the target $\widehat{w}^{(1)}$ to $w^{(1)} \in \mathbb{F}_q^{k_1+1}$ and let $Z := \{i_1 \mid w^{(1)}_{i_1} = 0\}$ denote the indices corresponding to the zero coordinates of the target.

2 Draw $j \in \{2,3\}$ uniformly and set $j' \in \{2,3\} \setminus j$.

3 Multiply out the polynomial $\prod_{i_1 \in Z}\left(\Lambda_{j'} - \lambda_{i_1,j'}\right) \in \mathbb{F}_q[\Lambda_{j'}]$ in the indeterminate $\Lambda_{j'}$ as

$$\sum_{i_{j'}=0}^{k} w^{(j')}_{i_{j'}} \Lambda^{i_{j'}}_{j'} \leftarrow \prod_{i_1 \in I_{j'}} \left(\Lambda_{j'} - \lambda_{i_1,j'}\right)$$

and read out the polynomial coefficients as the coordinate vector $w^{(j')}$.

4 Set $\bar{Z} \leftarrow \{1, 2, \ldots, k_1\} \setminus Z$. Read off the coordinate vector $w^{(j)}$ as the coefficients of the polynomial

$$\sum_{i_j=0}^{k} w^{(j)}_{i_j} \Lambda^{i_j}_j \leftarrow \sum_{i_1 \in \bar{Z}} \frac{\left(X_1^{-1} w^{(1)}\right)_{i_1}}{\prod_{i_{j'} \in Z}\left(\lambda_{i_1,j'} - \lambda_{i_{j'},j'}\right)} \prod_{i'_1 \in \bar{Z}, i'_1 \neq i_1} \frac{\Lambda_j - \lambda_{i'_1,j}}{\lambda_{i_1,j} - \lambda_{i'_1,j}},$$

where $\left(X_1^{-1} w^{(1)}\right)_{i_1}$ is the i_1-th coordinate of $X_1^{-1} w^{(1)}$.

5 Output the tuple (after projectivising its vectors) $\left(X_2^{-1} w^{(2)}, X_3^{-1} w^{(3)}\right)$.

Remark 1 (Preimage entropy in four or more dimensions). There may be applications demanding a large preimage sampling support for every target. To this end, in three dimensions, one can consider targets inside the Hamming ball $\mathcal{B}_{k_2+1}(\mathbb{P}^{k_1})$ or permute dimensions, as is done later in $\mathsf{SamplePre}^{\mathrm{3DB}}$. In four dimensions or more, even for targets in the sphere $\mathcal{S}(\mathbb{P}^{k_1})$, $\mathsf{SamplePre}$ draws from a support of size $\binom{k_3+k_4+\ldots+k_r}{k_3,k_4,\ldots,k_r}$, which is exponentially big in $k_3, k_4, \ldots, k_r$.

3.3 Domain samplers

We present two domain sampling algorithms $\mathsf{SampleDom}^{\mathrm{DB}}$, $\mathsf{SampleDom}^{\mathrm{3D}}$ and a third that is a mixture of the two. The first applies to doubly boundary formats in any dimension. The second applies to boundary formats in three dimensions.

Algorithm 4: $\mathsf{SampleDom}^{\mathrm{DB}}(\psi)$
Domain sampler for doubly boundary formats

Input: An r-dimensional tensor ψ of doubly boundary format $(k_1+1)\times(k_2+1)\times\ldots\times(k_r+1)$, where $k_1 = k_2+k_3+\ldots+k_r$ and $k_2 = k_3+k_4+\ldots+k_r$.

Output: A sample $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)$ from $\mathbb{P}^{k_2}\times\mathbb{P}^{k_3}\times\ldots\times\mathbb{P}^{k_3}$ such that $\hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) \in \mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$.

1 Draw $\left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$ uniformly from $\mathbb{F}_q^{k_3+1}\times\mathbb{F}_q^{k_4+1}\times\ldots\times\mathbb{F}_q^{k_r+1}$.

2 Draw a uniform subset $I_2 \subset \{1, 2, \ldots, k_1\}$ of indices in the first dimension of size $|I_2| = k_2 = k_3+k_4+\ldots+k_r$. Solve for $w^{(2)} \in \mathbb{F}_q^{k_2+1}$ such that

$$\sum_{\substack{0\le i_2\le k_2\\ \vdots\\ 0\le i_r\le k_r}} \psi_{i_1,i_2,\ldots,i_r} w^{(2)}_{i_2} w^{(3)}_{i_3}\ldots w^{(r)}_{i_r} = 0, \forall i_1 \in I_2. \tag{3.3}$$

With $\left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$ already fixed, this is a homogeneous $\mathbb{F}_q$-linear system with one more variable than there are constraints. Therefore, it has a solution space of dimension at least one. If the solution space is of dimension exactly one, draw a non zero $w^{(2)}$ from this solution space. Else, start again from step 1.

3 If $w_H\left(\hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)\right) < k_2+1$, start again from step 1.

4 Output the representative $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$ of $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)$.

Lemma 2. *Restrict Algorithm 4 to inputs ψ that are restricted isomorphic to some non degenerate Vandermonde-Weyman-Zelevinsky tensor. Then*

- *the rejection sampling in steps 2 and 3 of Algorithm 4 pass with probability close to 1, meaning the algorithm terminates in expected polynomial time.*
- *the output $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)$ of Algorithm 4 induces a uniform image $\mathfrak{h}_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)$ in $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$.*

Proof. Let $\psi = \phi\langle\Lambda\rangle^X$ be restricted isomorphic to some Vandermonde-Weyman-Zelevinsky tensor $\phi\langle\Lambda\rangle$ where Λ is a $(k_1+1)\times(r-1)$ matrix $\Lambda = (\lambda_{i_1,j})_{0\leq i_1\leq k_1, 2\leq j\leq r}$ over $\mathbb{F}_q$ such that for each column j, $i_1 \neq i_1'$ implies $\lambda_{i_1,j} \neq \lambda_{i_1',j}$. Keep the notation from Algorithm 4 with input $\phi\langle\Lambda\rangle^X$. In particular, $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$ is the representative of $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)$ that is output. Our insistence on $w^{(2)}$ satisfying the linear system 3.3 ensures

$$f_\psi^1\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)_{i_1} = 0, \quad \forall i_1 \in I_2.$$

Since $|I_2| \geq k_3 + k_4 + \ldots + k_r$,

$$\hat{f}_\psi^1\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) \in \mathcal{B}_{k_2+1}(\mathbb{P}^{k_1}).$$

Our further insistence on $w_H\left(\hat{f}_\psi^1\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right)\right) = k_2 + 1$ ensures

$$\hat{f}_\psi^1\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \ldots, \widehat{w}^{(r)}\right) \in \mathcal{S}_{k_2+1}(\mathbb{P}^{k_1}).$$

All that remains is to prove uniformity in $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$. The possible support $\bar{I}_2 := \{0, 1, \ldots, k_1\} \setminus I_2$ of the image $f_\psi^1\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$ is uniform among subsets of size $k_2 + 1$. Therefore, it suffices to prove that

$$\left(f_\psi^1\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)\right)_{i_1\in\bar{I}_2} = \left(\sum_{\substack{0\leq i_2\leq k_2 \\ \cdots \\ 0\leq i_r\leq k_r}} \psi_{i_1,i_2,\ldots,i_r} w_{i_2}^{(2)} w_{i_3}^{(3)} \ldots w_{i_r}^{(r)}\right)_{i_1\in\bar{I}_2} \tag{3.4}$$

is uniform in $\left(\mathbb{F}_q^*\right)^{k_2+1}$ conditioned on I_2, after implicitly fixing some ordering of $\bar{I}_2$. Inverting the trapdoor bases, $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$ maps to

$$\left(u^{(2)}, u^{(3)}, \ldots, u^{(r)}\right) := \left(X_2^{-1}w^{(2)}, X_3^{-1}w^{(3)}, \ldots, X_r^{-1}w^{(r)}\right).$$

Similar to the proof of lemma 1, for $2 \leq j \leq r$, consider the polynomials

$$P_j(\Lambda_j) := \sum_{i_j=0}^{k_j} u_{i_j}^{(j)} \Lambda_j^{i_j} \in \mathbb{F}_q[\Lambda_j]$$

in indeterminates Λ_j with the coordinates of $u^{(j)}$ as the coefficients. The linear system 3.3 transforms in to the diagonal one

$$P_2(\lambda_{i_1,2})P_3(\lambda_{i_1,3})\ldots P_r(\lambda_{i_1,r}) = 0, \ i_1 \in I_2,$$

solving for the coefficients of $P_2(\Lambda_2)$, for the chosen $P_3(\Lambda_3), P_4(\Lambda_4), \ldots, P_r(\Lambda_r)$. Let us pause to understand the constraints imposed by the rejection sampling

conditions in steps 2 and 3. The acceptance condition in step 2 that the linear system 3.3 has a solution space of dimension exactly one therefore translates to none of the polynomials $P_3(\Lambda_3), P_4(\Lambda_4), \ldots, P_r(\Lambda_r)$ having a zero in I_2. To pass the rejection sampling condition in step 3, the Hamming weight must be exactly k_2+1, which is equivalent to none of the polynomials $P_3(\Lambda_3), P_4(\Lambda_4), \ldots, P_r(\Lambda_r)$ having a zero in $\bar{I_2}$. In summary, the total rejection sampling of the algorithm is passed on the condition

$$P_j(\lambda_{i_1,j}) \neq 0, \forall j \in \{3,4,\ldots,r\}, \forall i_1 \in \{0,1,\ldots,k_1\}, \tag{3.5}$$

that the j-th polynomial does not have a zero on the j-th column of Λ. Getting back from our detour, the expression 3.4 that we would like to prove is uniform now takes the form

$$\left(P_2(\lambda_{i_1,2})P_3(\lambda_{i_1,3})\ldots P_r(\lambda_{i_1,r})\right)_{i_1 \in \bar{I_2}}, \tag{3.6}$$

up to a non zero multiple that comes from the i_1-th diagonal entry of X_1. Therefore, it suffices to prove that the expression 3.6 is uniform. One choice for $u^{(2)}$ that satisfies the linear system 3.3 for an already chosen $\left(w^{(3)}, w^{(4)}, \ldots, w^{(r)}\right)$ (and therefore $\left(u^{(3)}, u^{(4)}, \ldots, u^{(r)}\right)$) is $P_2(\Lambda_2) = \prod_{i'_1 \in I_2} \left(\Lambda_j - \lambda_{i'_1,2}\right)$. But this is the only choice (up to a non zero multiple), due to the rejection sampling in step 2 to only consider samples that give a one dimensional solution space to the linear system 3.3. Therefore, expression 3.6 takes the form

$$\left(\prod_{i'_1 \in I_2} \left(\lambda_{i_1,2} - \lambda_{i'_1,2}\right) P_3(\lambda_{i_1,3})P_4(\lambda_{i_1,4})\ldots P_r(\lambda_{i_1,r})\right)_{i_1 \in \bar{I_2}}. \tag{3.7}$$

For each $i_1 \in \bar{I_2}$, $\prod_{i'_1 \in I_2} \left(\lambda_{i_1,2} - \lambda_{i'_1,2}\right)$ is a non zero constant, since the first column of Λ has distinct entries. Therefore, it suffices to prove that

$$\left(P_3(\lambda_{i_1,3})P_4(\lambda_{i_1,4})\ldots P_r(\lambda_{i_1,r})\right)_{i_1 \in \bar{I_2}} \tag{3.8}$$

is uniform in $\left(\mathbb{F}_q^*\right)^{k_2+1}$. Partition $\bar{I_2}$ into disjoint subsets $I_2 = \bigcup_{j=3}^{r} I_r$ such that $I_3 = k_3 + 1$ and $|I_j| = k_j$ for all $j > 3$. Such a partition is always possible, since we are in a doubly boundary format and $|\bar{I_2}| = k_2 + 1 = k_3 + 1 + k_4 + \ldots + k_r$. Before the rejection sampling step, $\left(u^{(3)}, u^{(4)}, \ldots, u^{(r)}\right)$ is chosen uniformly from tuples of non zero vectors. Therefore, before rejection sampling

$$((P_3(\lambda_{i_1,3}))_{i_1 \in I_3}, (P_4(\lambda_{i_1,3}))_{i_1 \in I_4}, \ldots, (P_r(\lambda_{i_1,r}))_{i_1 \in I_4}) \tag{3.9}$$

is uniform in $\left(\mathbb{F}_q^{k_3+1} \setminus \mathbf{0}\right) \times \left(\mathbb{F}_q^{k_4} \setminus \mathbf{0}\right) \times \ldots \times \left(\mathbb{F}_q^{k_r} \setminus \mathbf{0}\right)$. Here, we used the fact that the polynomial evaluation map is uniform, as long as the number of evaluation points (which are distinct) is at most one greater than the degree. Post rejection sampling, the expression 3.9 is uniform in tuples of vectors $\left(\mathbb{F}_q^*\right)^{k_3+1} \times \left(\mathbb{F}_q^*\right)^{k_4} \times \ldots \times \left(\mathbb{F}_q^*\right)^{k_r}$ with no zero coordinates, by condition 3.5. Therefore, for each index $i_1 \in \bar{I_2}$ in expression 3.8, there is one evaluation, namely the $P_j(\lambda_{i_1,j})$ such that $i_1 \in I_j$, which is uniform in $\mathbb{F}_q^*$ and independent of every other evaluation in 3.9. Therefore, the expression 3.8 is indeed uniform in $\left(\mathbb{F}_q^*\right)^{k_2+1}$, as claimed. □

We next present the domain sampler $\mathsf{SampleDom}^{\mathrm{3D}}$ that only works in three dimensions, but works for every boundary format in three dimensions. It is complementary to $\mathsf{SampleDom}^{\mathrm{BD}}$ in three dimensions. While $\mathsf{SampleDom}^{\mathrm{BD}}$ solves for the vector in the second dimension to zero out the required zero coordinates in the image, $\mathsf{SampleDom}^{\mathrm{3D}}$ solves for the vector in the third dimension. Consequently, for three dimensional boundary formats $(2n+1)\times(n+1)\times(n+1)$, where they both apply, they have different domain sample distributions (conditioned on an image).

Algorithm 5: $\mathsf{SampleDom}^{\mathrm{3D}}(\psi)$
Domain sampler for three dimensional boundary formats

Input: A three dimensional tensor ψ of boundary format $(k_1+1)\times(k_2+1)\times(k_3+1)$, where $k_1 = k_2 + k_3$.
Output: A sample $(\widehat{w}^{(2)}, \widehat{w}^{(3)})$ from $\mathbb{P}^{k_2}\times\mathbb{P}^{k_3}$ such that $\mathfrak{h}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \in \mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$.

1 Draw $w^{(2)}$ uniformly at random from $\mathbb{F}_q^{k_2+1}$.

2 Draw a uniform subset $I_3 \subset \{1,2,\ldots,k_1\}$ of indices in the first dimension of size $|I_3| = k_3$. Solve for $w^{(3)} \in \mathbb{F}_q^{k_3+1}$ such that

$$\sum_{0\le i_2\le k_2}\ \sum_{0\le i_3\le k_3} \psi_{i_1,i_2,i_3} w^{(2)}_{i_2} w^{(3)}_{i_3} = 0, \forall i_1 \in I_3. \tag{3.10}$$

With $w^{(2)}$ already fixed, this is a homogeneous $\mathbb{F}_q$-linear system with one more variable than there are constraints. Therefore, it has a solution space of dimension at least one. If the solution space is of dimension exactly one, draw a non zero $w^{(3)}$ from this solution space. Else, start again from step 1.

3 If $w_H\left(\hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)\right) < k_2+1$, start again from step 1.

4 Output $\left(w^{(2)}, w^{(3)}\right)$ as a representative of $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)$.

Lemma 3. *Restrict Algorithm 5 to inputs ψ that are restricted isomorphic to some non degenerate Vandermonde-Weyman-Zelevinsky tensor. Then*

- *the rejection sampling in steps 2 and 3 of Algorithm 5 pass with probability close to 1, meaning the algorithm terminates in expected polynomial time.*
- *the output $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)$ of Algorithm 5 induces a uniform image $\mathfrak{h}_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)$ in $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$.*

Proof. Let $\psi = \phi\langle\Lambda\rangle^X$ be restricted isomorphic to some Vandermonde-Weyman-Zelevinsky tensor $\phi\langle\Lambda\rangle$ where Λ is a $(k_1+1)\times(2)$ matrix $\Lambda = (\lambda_{i_1,j})_{0\le i_1\le k_1, 2\le j\le 3}$ over $\mathbb{F}_q$ such that for each column j, $i_1 \neq i_1'$ implies $\lambda_{i_1,j} \neq \lambda_{i_1',j}$. Keep the notation from Algorithm 5 with input $\phi\langle\Lambda\rangle^X$. Let $\left(w^{(2)}, w^{(3)}\right)$ be the representative of $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)$ that is output. Our insistence on $w^{(3)}$ satisfying the linear system 3.10 ensures $f^1_\psi\left(w^{(2)}, w^{(3)}\right)_{i_1} = 0, \quad \forall i_1 \in I_3$. Since $k_1 - |I_3| = k_1 - k_3 = k_2 + 1$,

$\hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \in \mathcal{B}_{k_2+1}(\mathbb{P}^{k_1})$. Our further insistence on $w_H\left(\hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)\right) = k_2 + 1$ ensures

$$\hat{f}^1_\psi\left(\widehat{w}^{(2)}\right) \in \mathcal{S}_{k_2+1}(\mathbb{P}^{k_1}).$$

All that remains is to prove uniformity in $\mathcal{S}_{k_2+1}(\mathbb{P}^{k_1})$. The possible support $\bar{I_3} := \{0, 1, \ldots, k_1\} \setminus I_3$ of the image $f^1_\psi\left(w^{(2)}, w^{(3)}\right)$ is uniform among subsets of size $k_2 + 1$. Therefore, it suffices to prove that

$$\left(f^1_\psi\left(w^{(2)}, w^{(3)}\right)\right)_{i_1 \in \bar{I_3}} = \left(\sum_{0 \leq i_2 \leq k_2} \sum_{0 \leq i_3 \leq k_3} \psi_{i_1,i_2,i_3} w^{(2)}_{i_2} w^{(3)}_{i_3}\right)_{i_1 \in \bar{I_3}} \tag{3.11}$$

is uniform conditioned on I_3, after implicitly fixing some ordering of $\bar{I_3}$. Inverting the trapdoor bases, $\left(w^{(2)}, w^{(3)}\right)$ maps to $\left(u^{(2)}, u^{(3)}\right) := \left(X_2^{-1} w^{(2)}, X_3^{-1} w^{(3)}\right)$. Similar to the proof of lemma 1, for $2 \leq j \leq 3$, consider the polynomials

$$P_j(\Lambda_j) := \sum_{i_j=0}^{k_j} u^{(j)}_{i_j} \Lambda_j^{i_j} \in \mathbb{F}_q[\Lambda_j]$$

in indeterminates Λ_j with the coordinates of $u^{(j)}$ as the coefficients. The linear system 3.3 transforms in to the diagonal one

$$P_2(\lambda_{i_1,2}) P_3(\lambda_{i_1,3}) = 0, \ i_1 \in I_3,$$

solving for the coefficients of $P_3(\Lambda_3)$, for the chosen $P_2(\Lambda_2)$. The acceptance condition in step 2 that the linear system 3.10 has a solution space of dimension exactly one means $P_2(\Lambda_2)$ has no zeroes in I_3. To pass the rejection sampling condition in step 3, the Hamming weight must be exactly $k_2 + 1$, which means $P_2(\Lambda_2)$ has no zeroes in $\bar{I_3}$. In summary, the total rejection sampling of the algorithm is passed on the condition that $P_2(\Lambda_2)$ has no zeroes on the column $(\lambda_{i_1,2})_{0 \leq i_2 \leq k_1}$ of Λ. The expression 3.11 that we would like to prove is uniform now takes the form

$$\left(P_2(\lambda_{i_1,2}) P_3(\lambda_{i_1,3})\right)_{i_1 \in \bar{I_3}}, \tag{3.12}$$

up to a non zero multiple that comes from the i_1-th diagonal entry of X_1. Therefore, it suffices to prove that the expression 3.12 is uniform. The one and only choice for $u^{(3)}$ (up to a non zero multiple) that satisfies the linear system 3.10 for an already chosen $w^{(2)}$ (and therefore $u^{(2)}$) is

$$P_3(\Lambda_3) = \prod_{i'_1 \in I_3} \left(\Lambda_j - \lambda_{i'_1,3}\right).$$

Therefore, expression 3.12 takes the form

$$\left(P_2(\lambda_{i_1,2}) \prod_{i'_1 \in I_3} \left(\lambda_{i_1,3} - \lambda_{i'_1,3}\right)\right)_{i_1 \in \bar{I_3}}. \tag{3.13}$$

For each $i_1 \in \bar{I}_3$, $\prod_{i'_1 \in I_3} \left(\lambda_{i_1,3} - \lambda_{i'_1,3}\right)$ is non zero constant, since every column of Λ has distinct entries. Therefore, it suffices to prove that $(P_2(\lambda_{i_1,2}))_{i_1 \in \bar{I}_3}$ is uniform in $\left(\mathbb{F}_q^*\right)^{k_2+1}$. Before any rejection sampling, $u^{(2)}$ is chosen uniformly, therefore $\left((P_2(\lambda_{i_1,2}))_{i_1 \in \bar{I}_3}\right)$ is uniform in $\mathbb{F}_q^{k_3+1} \backslash \mathbf{0}$. Here, we used the fact that the polynomial evaluation map is uniform, as the number k_2+1 of evaluation points (which are distinct) is at most one greater than the degree k_2. Post rejection sampling, $(P_2(\lambda_{i_1,2}))_{i_1 \in \bar{I}_3}$ is uniform in $\left(\mathbb{F}_q^*\right)^{k_2+1}$ with no zero coordinates, which proves the claim. □

4 Preimage and domain sampler compatibility

We continue section 3 in this appendix due to lack of space, finishing off with two lemmas pairing preimage and domain samplers for use in encryption and signatures.

Lemma 4 (Preimage-Domain sampler agreement for encryption). *Let $\mathfrak{t} = (\Lambda, X^{-1})$ be a trapdoor in three dimensional doubly boundary format $(2k+1) \times (k+1) \times (k+1)$ with the associated public tensor $\psi = \phi\langle\Lambda\rangle^X$. For every target $\widehat{w}^{(1)} \in \mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$ on the Hamming sphere,* SamplePre *draws a unique preimage $(\widehat{u}^{(2)}, \widehat{u}^{(3)}) \leftarrow \mathsf{SamplePre}(\mathfrak{t}, X_1^{-1}\widehat{w}^{(1)})$. Likewise,* $\mathsf{SampleDom}^{3D}$ *draws a unique domain sample $(\widehat{w}^{(2)}, \widehat{w}^{(3)}) \leftarrow \mathsf{SampleDom}^{3D}(\psi)$ conditioned on its image $\mathfrak{h}_\psi(\widehat{w}^{(2)}, \widehat{w}^{(3)})$ being the target $\widehat{w}^{(1)}$. Further, these samples are identical, meaning $(\widehat{w}^{(2)}, \widehat{w}^{(3)}) = (\widehat{u}^{(2)}, \widehat{u}^{(3)})$.*

Proof. Recall the notation from the description of SamplePre with trapdoor $\mathfrak{t} = (\Lambda, X^{-1})$. The only source of randomness in SamplePre is in the allocation of index sets $I_3, I_4 \ldots, I_r \subseteq Z$ to account for all the zeroes to hit in the first dimension. When the target $\widehat{w}^{(1)}$ lies on the Hamming sphere $\mathcal{S}_{k+1}(\mathbb{P}^{2k})$ and we are in three dimensions, there is only once choice, to take I_3 to be the set of all indices to zero out. Therefore, restricted to three dimensions and targets on the sphere $\mathcal{S}_{k+1}(\mathbb{P}^{2k})$, SamplePre is deterministic. Further, this unique preimage is determined by the polynomial

$$\sum_{i_3=0}^{k_3} u_{i_3}^{(3)} \Lambda_3^{i_3} = \prod_{i_1 \in I_3} (\Lambda_3 - \lambda_{i_1,3})$$

encoding as coefficients the coordinate vector $u^{(3)}$ designed to zero out the I_3 indices. Given $X^{-1}w^{(1)}$ and $u^{(3)}$, $\widehat{u}^{(2)}$ is determined by linear constraints.

Switch to notation from the domain sampler $\mathsf{SampleDom}^{3\mathrm{D}}(\psi)$ with input $\psi = \phi\langle\Lambda\rangle^X$. In three dimensions, the sample

$$(\widehat{w}^{(2)}, \widehat{w}^{(3)}) \leftarrow \mathsf{SampleDom}^{3\mathrm{D}}(\psi) \mid \mathfrak{h}_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) = w^{(1)}$$

drawn by $\mathsf{SampleDom}^{3\mathrm{D}}$ conditioned on the image of the sample being $\widehat{w}^{(1)}$, is unique. That is, $\mathsf{SampleDom}^{3\mathrm{D}}$ is deterministic conditioned on the image of the

sample it draws. This is because conditioned on the image being $\widehat{w}^{(1)}$, the only randomness in $\mathsf{SampleDom}^{\mathrm{3D}}$ comes from the choice of I_3, which is unique in three dimensions. Here again, the polynomial corresponding to the sample $\widehat{w}^{(3)}$ drawn by $\mathsf{SampleDom}^{\mathrm{3D}}$ in dimension 3 is $\prod_{i_1 \in I_3} (\Lambda_3 - \lambda_{i_1,3})$, up to a non zero constant multiple, thereby proving the lemma. □

For three dimensional doubly boundary formats $(2k+1) \times (k+1) \times (k+1)$, both domain samplers $\mathsf{SampleDom}^{\mathrm{DB}}$ and $\mathsf{SampleDom}^{\mathrm{3D}}$ apply. We define a third sampler $\mathsf{SampleDom}^{\mathrm{3DB}}$ for such formats that flips a fair coin and calls one of $\mathsf{SampleDom}^{\mathrm{DB}}$ and $\mathsf{SampleDom}^{\mathrm{3D}}$ randomly.

Algorithm 6: $\mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$
Domain sampler for three dimensional doubly boundary formats

Input: A three dimensional tensor ψ of doubly boundary format $(2k+1) \times (k+1) \times (k+1)$.
Output: A sample $(\widehat{w}^{(2)}, \widehat{w}^{(3)})$ from $\mathbb{P}^k \times \mathbb{P}^k$ with $\hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \in \mathcal{S}_{k+1}(\mathbb{P}^k)$.

1 Flip a fair coin. If heads, call $\mathsf{SampleDom}^{\mathrm{DB}}(\psi)$. Else, call $\mathsf{SampleDom}^{\mathrm{3D}}(\psi)$.

Lemma 5 (Preimage-Domain sampler indistinguishability for signature simulation). *Let $\mathfrak{t} = (\Lambda, X^{-1})$ be a trapdoor in three dimensional doubly boundary format $(2k+1) \times (k+1) \times (k+1)$ with the associated public tensor $\psi = \phi\langle\Lambda\rangle^X$. The distribution of $\mathsf{SampleDom}^{3DB}(\psi)$ conditioned on a target $\widehat{w}^{(1)}$ and the corresponding preimage sample $\mathsf{SamplePre}^{3DB}(\mathfrak{t}, X_1^{-1}\widehat{w}^{(1)})$ are the same. Further, the common underlying distribution is the uniform distribution on two preimage choices.*

Proof. Rethink of $\mathsf{SamplePre}^{\mathrm{3DB}}$ as follows. The coin toss in step 2 chooses whether or no to permute dimensions 2 and 3. Then $\mathsf{SamplePre}$ is run. On the other side, $\mathsf{SampleDom}^{\mathrm{3DB}}$ can also be reinterpreted as follows. The coin toss in step 1 chooses whether or no to permute dimensions 2 and 3. Then $\mathsf{SampleDom}^{\mathrm{3D}}$ is run. Therefore the distribution of $\mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$ conditioned on a target $\widehat{w}^{(1)}$ and the preimage sampler $\mathsf{SamplePre}^{\mathrm{3DB}}(\mathfrak{t}, \widehat{w}^{(1)})$ are the same.

All that remains is to determine the common distribution. To this end, let us look at $\mathsf{SamplePre}^{\mathrm{3DB}}$ again. We claim that each choice $j \in \{2,3\}$ of the coin toss results in a different preimage. This can be verified by inspecting in which dimension the polynomial interpolated in step 3 is placed. If the two choices $j \in \{2,3\}$ result in the same preimage, then the image $\widehat{w}^{(1)}$ will have $2k$ zero coordinates, placing it far in the interior of the sphere $\mathcal{S}_{k+1}\left(p^{2k}\right)$, a contradiction. Therefore, for every $\widehat{w}^{(1)}$, the preimages $\mathsf{SamplePre}^{\mathrm{3DB}}\left(\mathfrak{t}, \widehat{w}^{(1)}\right)$ are drawn uniformly from two distinct choices. □

We conclude the section with a quick cross reference to the various preimage and domain samplers, foreshadowing their later use.

	SamplePre	$\mathsf{SamplePre}^{\mathrm{3DB}}$
SampleDom	Indistinguishable conditioned on a target. Used in Σ^{DB} signatures.	
$\mathsf{SampleDom}^{\mathrm{3D}}$	Exact agreement conditioned on a target. Used in Π^{3DB} encryption.	
$\mathsf{SampleDom}^{\mathrm{3DB}}$: a mixture of SampleDom and $\mathsf{SampleDom}^{\mathrm{3D}}$		Identical distributions conditioned on a target. Used in Σ^{3DB} signatures.

Table 1. A menagerie of preimage and domain samplers.

4.1 Hardness assumptions

We make two cryptographic hardness assumptions in the security proofs of the cryptographic primitives based on the trapdoor one-way function construction. The first concerns the pseudorandomenss of Vandermonde-Weyman-Zelevinsky tensor orbits, and is closely related to standard tensor isomorphism hardness assumptions. In spirit, the assumption is that the public tensors generated by our trapdoor generator is computationally indistinguishable from random tensors of the same format. The second assumption is the hardness of solving random multilinear equations over finite fields; which we formulate as a collision resistance assumption. This translates to collision finding of the one way function h_ψ corresponding to a truly random tensor ψ being hard. We justify the assumptions by benchmarking using the best known algorithms for these or closely related problems. A thorough algorithmic analysis of the hardness focused on our assumptions is warranted, which we leave to future work.

The discussion on hardness will focus on three and four dimensions. Since the hash function description sizes grow exponentially with dimension, the utility of going to higher dimensions is unclear, unless the lengths in all but the first dimension are small. But there is evidence that in high dimensions and small lengths, the preimage sampling problem may be solved without the trapdoor in subexponential time using hyperdeterminants [23]. In both three and four dimensions, we will focus on doubly boundary formats. In three dimensions, these are $(2k+1) \times (k+1) \times (k+1)$ formats. In four dimensions, these are of the form $(2(k_3+k_4)+1) \times (k_3+k_4+1) \times (k_3+1) \times (k_4+1)$. There might be applications, where breaking the symmetry of the format by asking each dimension has a different length may be useful. For instance, we may want to elude some tensor isomorphism cryptanalytic algorithms [26] that exploit the symmetry among dimensions. But for simplicity, we will take the one parameter family of four dimensional formats $(4k+1) \times (2k+1) \times (k+1) \times (k+1)$ as a proxy for formats where the last two dimensions have roughly the same length.

Pseudorandomness of Vandermonde-Weyman-Zelevinsky orbits. Keeping the notation from the trapdoor generator $\mathsf{TrapGen}$, the union of all restricted isomorphism orbits of non-singular Vandermonde-Weyman-Zelevinsky tensors is $\bigcup_{\Lambda,X}\{\phi\langle\Lambda\rangle^X\}$, with some format implicitly in mind. In particular, the union is over the trapdoors $\mathsf{t} = (\Lambda, X^{-1})$ that $\mathsf{TrapGen}$ draws from.

Pseudorandomness of Vandermonde-Weyman-Zelevinsky orbits assumption: Consider either the tensor format family $(2k+1)\times(k+1)\times(k+1)$ or $(4k+1)\times(2k+1)\times(k+1)\times(k+1)$ over a family of finite fields $\mathbb{F}_q$. To randomized polynomial (in k and $\log q$) time algorithms, a random tensor in the union $\bigcup_{\Lambda,X}\{\phi\langle\Lambda\rangle^X\}$ of orbits of non-singular Vandermonde-Weyman-Zelevinsky tensors is indistinguishable from a truly random tensor of the same format.

The assumption is closely related to those made in tensor isomorphism based cryptography, such as the pseudorandomness assumptions [36][Conjecture 1], [7][Section 6.1] and [22][Section 4]. They assume that a pair of random isomorphic tensors is computationally indistinguishable from a pair of random tensors. The notion of isomorphism may vary with context, but the most relevant to us is the general linear action of matrices acting in each dimension by multiplication. While these assumptions and ours are closely related, they are incomparable with the following distinctions. Our seems like the safer assumption in the sense that we only present one tensor to the indistinguishability game, not a pair of tensors. In particular, algorithms for finding tensor isomorphisms do not directly apply to our case, since they would not know which one of exponentially many Vandermonde-Weyman-Zelevinsky tensors to compare to. In other senses, our assumption seems a little more delicate. For one, the matrix acting in the first dimension in our case is a diagonal matrix and not a uniform invertible matrix. This issue is only in three dimensions. In four dimensions, the triple of uniform invertible matrices acting in dimensions two through four should make our assumption safer. In particular, our four dimensional assumption can be expressed as a set of three dimensional assumptions, by taking slices in the first dimension. The other, greater cause for concern is that the isomorphism classes we are hiding have algebraic structure. One may formulate our pseudorandom assumption as testing if a multivariate polynomial system with the entries of the matrices $\Lambda, X_1, X_2, \ldots, X_r$ as the variables has a solution. This should take time exponential in k^2, judging from just the number of variables, the number of constraints and the degree of the system. We leave the empirical and theoretical analysis of such a system using Gröbner bases to future work.

Hardness of solving random multilinear equations over finite fields. Our second assumption is that for a random boundary format tensor ψ, it is hard to invert the one-way function $\mathfrak{h}_\psi$. Note that the assumption concerns random tensors, and is not restricted to tensors isomorphic to some Vandermonde-Weyman-Zelevinsky tensor. Therefore, there is no structure to the tensors in the

assumption for one to exploit. We next write it down explicitly for three and four dimensions.

Definition 9 (Bilinear system solving). *The bilinear system solving problem is given a tensor ψ of format $(2k+1)\times(k+1)\times(k+1)$ over a finite field $\mathbb{F}_q$ and a target $\widehat{w}^{(1)} \in \mathbb{P}^{2k}$, to find a pair of projective vectors $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \in \mathbb{P}^k \times \mathbb{P}^k$ such that $\hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) = \widehat{w}^{(1)}$.*

Bilinear system solving hardness assumption: The bilinear system solving problem for random tensors ψ restricted to targets in the Hamming sphere $\mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$ takes at least $2^k(\log q)^{O(1)}$ time.

Definition 10 (Trilinear system solving). *The trilinear system solving problem is given a tensor ψ of format $(4k+1)\times(2k+1)\times(k+1)\times(k+1)$ over a finite field $\mathbb{F}_q$ and a target $\widehat{w}^{(1)} \in \mathbb{P}^{4k}$, to find a triple of projective vectors $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \widehat{w}^{(4)}\right) \in \mathbb{P}^k \times \mathbb{P}^k \times \mathbb{P}^k$ such that $\hat{f}^1_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}, \widehat{w}^{(4)}\right) = \widehat{w}^{(1)}$.*

Trilinear system solving hardness assumption: The trilinear system solving problem for random tensors ψ restricted to targets in the Hamming sphere $\mathcal{S}_{2k+1}\left(\mathbb{P}^{4k}\right)$ takes at least $4^k(\log q)^{O(1)}$ time.

We next discuss the plausibility of the bilinear system solving hardness assumption, taking for granted that the trilinear version is even more plausible. Emiris, Mantzaflaris, and Tsigaridas [11] study the computational complexity of the bilinear system solving problem in almost exactly the same formulation as above, with the following differences:

- Their underlying field is the real numbers, instead of our finite fields. But their findings should translate to the finite field setting since the methods are algebraic (using Sylvester determinants, Weyman resultant complexes, etc.) without involving analysis.
- Their formulation is in terms of a set of matrices (for instance [11][equation 1]) instead of a tensor, but this is easy to translate by seeing their matrices as slices of our tensor in the first dimension.
- Their target is from the Hamming sphere $\mathcal{S}_1(\mathbb{P}^{k_2})$ of radius one, in contrast to our much larger sphere. Still their setting is representative of the difficulty of our problem, since there are a lot of zeros in both the targets.
- Their system needs to be zero dimensional and their complexity bounds are in terms of the number of solutions of the zero dimensional system. Since zero dimensionality is the generic case, it holds for random tensors with high probability. Being in a boundary format makes these generic systems zero dimensional.

With these differences in mind, the dominant term in their algorithm's run time (ignoring precision issues) is $\mathcal{O}\left(k^4\binom{2k}{k}^4\right)$. Here, the binomial $\binom{2k}{k}$ term appears due to the bivariate Bezout bound, accounting for the number of solutions. For the random ψ in non-singular orbits that appear as inputs in our constructions, a similar bound is indeed met, since we can choose from exponentially many ways

to distribute the zeroes between the two dimensions. In light of the binomial $\binom{2k}{k}$ growing at least as 2^k, our Assumption 2 (Bilinear) seems conservative and plausible. See also Spaenlehauer's works [33,34] for similar evidence using Gröbner basis and other techniques. Finally, guided by this evidence we make the following seemingly stronger collision resistance assumption.

Definition 11 (Bilinear collision finding). *The bilinear collision finding problem is given a tensor ψ of format $(2k+1)\times(k+1)\times(k+1)$ over a finite field $\mathbb{F}_q$, to find two distinct pairs of projective vectors $\left(\widehat{w}^{(2)},\widehat{w}^{(3)}\right),\left(\widehat{u}^{(2)},\widehat{u}^{(3)}\right)\in\mathbb{P}^k\times\mathbb{P}^k$ such that $\hat{f}^1_\psi\left(\widehat{w}^{(2)},\widehat{w}^{(3)}\right)=\hat{f}^1_\psi\left(\widehat{u}^{(2)},\widehat{u}^{(3)}\right)$ and $\hat{f}^1_\psi\left(\widehat{w}^{(2)},\widehat{w}^{(3)}\right)\in\mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$.*

Bilinear collision finding hardness assumption: The bilinear collision finding problem for random tensors ψ takes at least $2^k(\log q)^{O(1)}$ time.

Parameter selection. Let $\mathsf{ParamSel}(1^\lambda, r)$ denote a parameter selection algorithm that given a desired security parameter 1^λ and dimension, outputs the sequence of positive numbers $(q, k_1, k_2, \ldots, k_r)$ to signify that our scheme is over the finite field $\mathbb{F}_q$ and built with tensors of format $(k_1+1)\times(k_2+1)\times\ldots\times(k_r+1)$. The field size and tensor format are chosen such that the underlying hard problems take at least 2^λ time to break. Guided by the hardness assumptions, for three and four dimensions, we may instantiate $\mathsf{ParamSel}$ as follows. Set $\mathsf{ParamSel}(1^\lambda, 3)$ to output $(q, 4\lambda, 2\lambda, 2\lambda)$ where q is the smallest prime number greater than 4λ. Thereby, there are at least $\binom{4\lambda}{2\lambda+1}$ (which is, exponentially) many choices for each column of the matrix Λ defining our trapdoors. Likewise, for four dimensions, set $\mathsf{ParamSel}(1^\lambda, 4)$ to output $(q, 8\lambda, 4\lambda, 2\lambda, 2\lambda)$ where q is the smallest prime number greater than 8λ.

5 Hash-and-Sign signatures from tensors

We next present a simple Hash-and-Sign signature scheme from tensors, closely following the GPV recipe in both design and proof. With the format implicitly in mind, fix a cryptographic hash function

$$\mathsf{H}:\{0,1\}^*\longrightarrow\mathcal{S}_{k_2+1}\left(\mathbb{P}^{k_1}\right).$$

The Hash-and-Sign signature scheme Σ^{DB} in r-dimensional doubly boundary formats consists of the following key generator, signing and verification algorithms.

$\Sigma^{\mathrm{DB}}.\mathsf{KeyGen}(1^\lambda)$

$(q, k_1, k_2, \ldots, k_r) \leftarrow \mathsf{ParamSel}(1^\lambda, r)$
$(\psi, \mathsf{t}) \leftarrow \mathsf{TrapGen}\left(q, 1^{k_1}, 1^{k_2}, \ldots, 1^{k_r}\right)$
return $\mathsf{vk} = \psi, \mathsf{sk} = \mathsf{t}$

$\Sigma^{\mathrm{DB}}.\mathsf{Sign}(\mathsf{sk}, \mu)$

$s \leftarrow\!\!\$\ \{0,1\}^\lambda$
$\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) \leftarrow \mathsf{SamplePre}(\mathsf{t}, \mathsf{H}(\mu, s))$
$\sigma \leftarrow \left(\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right), s\right)$
return σ

$\Sigma^{\mathrm{DB}}.\mathsf{Verify}(\mathsf{vk}, \sigma, \mu)$

return $\mathsf{H}(\mu, s) \stackrel{?}{=} \mathfrak{h}_\psi\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$

Fig. 3. Σ^{DB}: Hash-and-Sign signatures from doubly boundary format tensors

The key generation algorithm KeyGen invokes the trapdoor generator $\mathsf{TrapGen}$ and outputs the tensor-trapdoor pain ψ, t (corresponding to the one-way function-trapdoor pair $(\mathfrak{h}_\psi, \mathsf{t})$) received, as the verification-signing key pair $(\mathsf{vk}, \mathsf{sk})$.

The signing algorithm Sign takes a message μ and the secret key sk as inputs. It draws a uniform λ long bit string as salt s and computes of the hash $\mathsf{H}(\mu, s)$ of the salted message. The salt is this long to allow roughly $2^{\lambda/2}$ signature queries without worrying about message-salt collisions. If one only cares about polynomial time adversaries, the salt length can be reduced to being just super logarithmic $\omega(\log \lambda)$ in the security parameter. This hash value $\mathsf{H}(\mu, s)$ is a Hamming weight $k_2 + 1$ projective vector lying in $\mathcal{S}_{k_2+1}\left(\mathbb{P}^{k_1}\right)$, by design. From the secret key sk, the signing algorithm knows the trapdoor $\mathsf{t} = (\Lambda, X^{-1})$ made up of the secret matrix generating the Vandermonde-Weyman-Zelevinsky tensor Λ and the secret basis change X^{-1}. With this information at hand, it calls the preimage sampling algorithm $\mathsf{SamplePre}$ to compute a preimage $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$ and outputs the preimage-salt pair as the signature $\sigma = \left(\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right), s\right)$. For a signature generated by an honest signer,

$$\mathsf{H}(\mu, s) = \mathfrak{h}_\psi\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right),$$

which the verification algorithm Verify duly verifies, for the preimage-salt pair in the claimed signature. Therefore, the completeness of the protocol is clear, in that honestly generated signatures always pass verification.

Remark 2. In certain cases where two dimensions of the format are of the same length, a valid signature corresponding to a public verification key ψ may be copied and claimed as a valid signature corresponding to another public verification key ψ obtained by permuting those equal length dimensions. If an application demands such attacks be prevented, we can append the public key to

the argument of the hash function. In particular, modify the second line of the signing algorithm to $\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right) \leftarrow \mathsf{SamplePre}(\mathfrak{t}, \mathsf{H}(\psi, \mu, s))$ and the only line of the verification algorithm to $\mathsf{H}(\psi, \mu, s) \stackrel{?}{=} \mathfrak{h}_\psi\left(w^{(2)}, w^{(3)}, \ldots, w^{(r)}\right)$. Doing so commits the public key to the signatures and prevents such attacks. This however comes at the cost of slowing down the signature algorithm, which otherwise does not explicitly have to write down tensors.

We next present the Hash-and-Sign signature scheme Σ^{3D} for three dimensional doubly boundary formats $(2k+1) \times (k+1) \times (k+1)$. The scheme is not merely a specialisation of Σ^{DB} to three dimensions. Instead, it carefully deploys the preimage sampler $\mathsf{SamplePre}^{\mathrm{3DB}}$, whose statistics can be matched (conditioned on a target) by that of the domain sampler $\mathsf{SampleDom}^{\mathrm{3DB}}$ in the signature simulation phase of the security proof.

$\Sigma^{\mathrm{3DB}}.\mathsf{KeyGen}(1^\lambda)$

$(q, 2k, k, k) \leftarrow \mathsf{ParamSel}(1^\lambda, 3)$
$(\psi, \mathfrak{t}) \leftarrow \mathsf{TrapGen}\left(q, 1^{2k}, 1^k, 1^k\right)$
return $\mathsf{vk} = \psi, \mathsf{sk} = \mathfrak{t}$

$\Sigma^{\mathrm{3DB}}.\mathsf{Sign}(\mathsf{sk}, \mu)$

$s \leftarrow\!\!\$\ \{0,1\}^\lambda$
$\left(w^{(2)}, w^{(3)}\right) \leftarrow \mathsf{SamplePre}^{\mathrm{3DB}}(\mathfrak{t}, \mathsf{H}(\mu, s))$
$\sigma \leftarrow \left(\left(w^{(2)}, w^{(3)}\right), s\right)$
return σ

$\Sigma^{\mathrm{3DB}}.\mathsf{Verify}(\mathsf{vk}, \sigma, \mu)$

return $\mathsf{H}(\mu, s) \stackrel{?}{=} \mathfrak{h}_\psi\left(w^{(2)}, w^{(3)}\right)$

Fig. 4. Σ^{3DB}: Hash-and-Sign signatures from 3D doubly boundary format tensors

5.1 Unforgeability analysis in the ROM

We next present the Q-query EUF-CMA$_Q$ security analysis for Σ^{3DB} in the random oracle model and show that an adversary forging signatures can break one of our hardness assumptions. Consider an adversary $\mathcal{A}$ playing the Q-query EUF-CMA$_Q$ game pictured below in figure 5. Define the advantage of the adversary playing the Q-query EUF-CMA$_Q$ game as

$$\mathsf{Adv}^{\text{EUF-CMA}}_{\mathcal{A}, \Sigma^{\text{3DB}}, Q}(\lambda) := \Pr\left[\text{EUF-CMA}^{\mathcal{A}}_{\Sigma^{\text{3DB}}, Q}(\lambda) = 1\right].$$

$\text{EUF-CMA}^{\mathcal{A}}_{\Sigma^{\text{3DB}},Q}(\lambda)$	$\mathsf{SIGN}(\mu)$	$\mathsf{RO}(\mu, s)$
$\mathcal{Q}, \mathcal{H}, c \leftarrow \emptyset, \emptyset, 0$	**if** $c \geq Q$ **then abort**	**if** $(\mu, s) \notin \mathcal{H}$ **then**
$\mathsf{vk}, \mathsf{sk} \leftarrow \Sigma^{\text{3DB}}.\mathsf{KeyGen}(1^\lambda)$	$\sigma \leftarrow \Sigma^{\text{3DB}}.\mathsf{Sign}(\mathsf{sk}, \mu)$	$\mathcal{H}[\mu, s] \leftarrow\!\!\$\ \mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$
$(\mu^\star, \sigma^\star) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}(\mathsf{vk})$	$c \leftarrow c + 1$	**return** $\mathcal{H}[\mu, s]$
return $(\mu^\star, \cdot) \notin \mathcal{Q} \wedge \Sigma^{\text{3D}}.\mathsf{Verify}(\mathsf{vk}, \sigma^\star, \mu^\star)$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$	
	return σ	

Fig. 5. Q query EUF-CMA_Q in the ROM for Σ^{3DB}.

Theorem 3 (Unforgeability). *Assume the pseudorandomness of Vandermone-Weyman-Zelevinsky tensor orbits assumption, as described in section 4.1. The advantage* $\mathsf{Adv}^{\text{EUF-CMA}}_{\mathcal{A},\Sigma^{3DB},Q}(\lambda)$ *of any Q-query bounded probabilistic polynomial time adversary $\mathcal{A}$ playing the* EUF-CMA_Q *security game in the ROM game pictured in figure 5 is bounded as*

$$\mathsf{Adv}^{\text{euf-cma}}_{\mathcal{A},\Sigma,Q}(\lambda) \leq 2\ \mathsf{Adv}^{\text{bilinear}-\text{collision}}_{2k+1,k+1,k+1}(\lambda) + Q\ \mathsf{negl}(\lambda)$$

where,

- $\mathsf{Adv}^{\text{bilinear}-\text{collision}}_{2k+1,k+1,k+1}(\lambda)$ *is the advantage of any randomized polynomial time adversary in solving the bilinear collision finding defined in section 4.1*
- $\mathsf{negl}(\lambda)$ *is an exponentially small function in λ.*

Proof. We prove the theorem through a game hop analysis. Let $\Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game_i}}(\psi) = 1\right]$ the success probability of the adversary playing the i-th game $\mathsf{Game_i}$. We start

Game_0	$\mathsf{SIGN}(\mu)$	$\mathsf{RO}(\mu, s)$
$\mathcal{Q}, \mathcal{H}, c \leftarrow \emptyset, \emptyset, 0$	**if** $c \geq Q$ **then abort**	**if** $(\mu, s) \notin \mathcal{H}$ **then**
$(q, 2k, k, k) \leftarrow \mathsf{ParamSel}(1^\lambda, r)$	$s \leftarrow\!\!\$\ \{0,1\}^\lambda$	$\mathcal{H}[\mu, s] \leftarrow\!\!\$\ \mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$
$(\psi, \mathfrak{t}) \leftarrow \mathsf{TrapGen}\left(q, 1^{2k}, 1^k, 1^k\right)$	$\sigma \leftarrow (\mathsf{SamplePre}^{\text{3DB}}(\mathfrak{t}, \mathsf{RO}(\mu, s)), s)$	**return** $\mathcal{H}[\mu, s]$
$(\mu^\star, \widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}, s^*) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}(\psi)$	$c \leftarrow c + 1$	
$b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$	
$b_1 := \mathcal{H}[\mu^\star, \boldsymbol{r}^\star] \stackrel{?}{=} \mathfrak{h}_\psi\left(\widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}\right)$	**return** σ	
return $b_0 \wedge b_1$		

Fig. 6. Game 0.

with Game_0 above, merely rephrasing the Q query EUF-CMA_Q in the ROM,

Game_1	$\mathsf{SIGN}(\mu)$	$\mathsf{RO}(\mu, s)$
$\mathcal{Q}, \mathcal{H}, \mathcal{P}, c \leftarrow \emptyset, \emptyset, 0$	**if** $c \geq Q$ **then abort**	**if** $(\mu, s) \notin \mathcal{H}$ **then**
$(q, 2k, k, k) \leftarrow \mathsf{ParamSel}(1^\lambda, 3)$	$s \leftarrow\!\!\$\ \{0,1\}^\lambda$	$\mathcal{P}[\mu, s] \leftarrow \mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$
$(\psi, \mathfrak{t}) \leftarrow \mathsf{TrapGen}\left(q, 1^{2k}, 1^k, 1^k\right)$	$\sigma \leftarrow (\mathsf{SamplePre}^{\mathfrak{t},\mathrm{3DB}}(\mathsf{RO}(\mu, s)), s)$	$\mathcal{H}[\mu, s] \leftarrow \mathfrak{h}_\psi(\mathcal{P}[\mu, s])$
$(\mu^\star, \widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}, s^*) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}(\psi)$	$c \leftarrow c + 1$	**return** $\mathcal{H}[\mu, s]$
$b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$	
$b_1 := \mathcal{H}[\mu^\star, s^\star] \stackrel{?}{=} \mathfrak{h}_\psi\left(\widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}\right)$	**return** σ	
return $b_0 \wedge b_1$		

Fig. 7. Game 1. RO Programming

specialised to Σ^{3DB}. In the first hop, we program the random oracle RO to answer hash. If the message-salt pair has already been queried before, we look up the list of such queries and answer consistently, by returning the same output that is stored in the list of images. If the message-salt pair has never been queried before, we call the domain sampler $\mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$ on the public tensor ψ to draw a preimage-image pair. The preimage is stored in the list of preimages and the image is stored in the list of images. Both lists are indexed by message-salt pairs. By lemma 2 and lemma 3, irrespective of the outcome of the coin flip in $\mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$, the image $\mathfrak{h}_\psi(\mathsf{SampleDom}^{\mathrm{3DB}}(\psi))$ is uniform in $\mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$. Therefore, Game_0 and Game_1 induce identical distributions from the view of the adversary, meaning

$$\Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game}_0}(\psi) = 1\right] = \Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game}_1}(\psi) = 1\right].$$

In the second game hop, we detect within the signing subroutine if the random oracle is queried with an already queried message-salt. If so, we label it as a bad event and abort. By the fundamental lemma of game playing, the probability of such a repeated message-salt query is at most $Q/2^{\lambda/2}$. Therefore,

$$\left|\Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game}_1}(\psi) = 1\right] - \Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game}_2}(\psi) = 1\right]\right| \leq \frac{Q}{2^{\lambda/2}} = Q\ \mathsf{negl}(\lambda).$$

In the hop to Game_3, the signing algorithm is modified to simulate signatures as follows. For the message-salt (μ, s) pair under consideration, instead of setting to signature to $(\mathsf{SamplePre}^{\mathrm{3DB}}_{\mathfrak{t}}(\mathsf{RO}(\mu, s)), s)$, call the random oracle to get $\mathsf{RO}(\mu, s)$, and look up the preimage $\mathcal{P}[\mu, s]$ stored under the index (μ, s). Then, set $(\mathcal{P}[\mu, s], s)$ as the signature. By lemma 5, the output of $\mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$ conditioned on its image $\mathfrak{h}_\psi\left(\mathsf{SampleDom}^{\mathrm{3DB}}(\psi)\right)$ being some target $\widehat{w}^{(1)} \in \mathcal{S}_{k+1}\left(\mathbb{P}^k\right)$ is exactly the same as the distribution of $\mathsf{SamplePre}^{\mathrm{3DB}}(\mathfrak{t}, \widehat{w}^{(1)})$. Therefore, the adversary views for Game_2 and Game_3 are identical and

$$\Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game}_2}(\psi) = 1\right] = \Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game}_3}(\psi) = 1\right].$$

Game_2	$\mathsf{SIGN}(\mu)$	$\mathsf{RO}(\mu, s)$
$\mathcal{Q}, \mathcal{H}, \mathcal{P}, c \leftarrow \emptyset, \emptyset, \emptyset, 0$	**if** $c \geq Q$ **then abort**	**if** $(\mu, s) \notin \mathcal{H}$ **then**
$repeat \leftarrow \mathsf{false}$	$s \leftarrow\!\!\$\ \{0,1\}^\lambda$	$\mathcal{P}[\mu, s] \leftarrow \mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$
$(q, 2k, k, k) \leftarrow \mathsf{ParamSel}(1^\lambda, 3)$	**if** $(\mu, s) \in \mathcal{H}$	$\mathcal{H}[\mu, s] \leftarrow \mathfrak{h}_\psi(\mathcal{P}[\mu, s])$
$(\psi, \mathfrak{t}) \leftarrow \mathsf{TrapGen}\left(q, 1^{2k}, 1^k, 1^k\right)$	$repeat \leftarrow \mathsf{true}$	**return** $\mathcal{H}[\mu, s]$
$(\mu^\star, \widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}, s^*) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}(\psi)$	**abort**	
$b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$	$\sigma \leftarrow (\mathsf{SamplePre}^{\mathrm{3DB}}(\mathfrak{t}, \mathsf{RO}(\mu, s)), s)$	
$b_1 := \mathcal{H}[\mu^\star, s^\star] \stackrel{?}{=} \mathfrak{h}_\psi\left(\widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}\right)$	$c \leftarrow c + 1$	
return $b_0 \wedge b_1$	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$	
	return σ	

Fig. 8. Game 2. Collision avoidance

Game_3	$\mathsf{SIGN}(\mu)$	$\mathsf{RO}(\mu, s)$
$\mathcal{Q}, \mathcal{H}, \mathcal{P}, c \leftarrow \emptyset, \emptyset, \emptyset, 0$	**if** $c \geq Q$ **then abort**	**if** $(\mu, s) \notin \mathcal{H}$ **then**
$repeat \leftarrow \mathsf{false}$	$s \leftarrow\!\!\$\ \{0,1\}^\lambda$	$\mathcal{P}[\mu, s] \leftarrow \mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$
$(q, 2k, k, k) \leftarrow \mathsf{ParamSel}(1^\lambda, 3)$	**if** $(\mu, s) \in \mathcal{H}$	$\mathcal{H}[\mu, s] \leftarrow \mathfrak{h}_\psi(\mathcal{P}[\mu, s])$
$(\psi, \mathfrak{t}) \leftarrow \mathsf{TrapGen}\left(q, 1^{2k}, 1^k, 1^k\right)$	$repeat \leftarrow \mathsf{true}$	**return** $\mathcal{H}[\mu, s]$
$(\mu^\star, \widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}, s^*) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}(\psi)$	**abort**	
$b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$	$\mathsf{RO}(\mu, s)$	
$b_1 := \mathcal{H}[\mu^\star, s^\star] \stackrel{?}{=} \mathfrak{h}_\psi\left(\widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}\right)$	$\sigma \leftarrow (\mathcal{P}[\mu, s], s)$	
return $b_0 \wedge b_1$	$c \leftarrow c + 1$	
	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$	
	return σ	

Fig. 9. Game 3. Signature simulation

The final game hop forgets the trapdoor generation and in its place draws

Game_4	$\mathsf{SIGN}(\mu)$	$\mathsf{RO}(\mu, s)$
$\mathcal{Q}, \mathcal{H}, \mathcal{P}, c \leftarrow \emptyset, \emptyset, \emptyset, 0$	**if** $c \geq Q$ **then abort**	**if** $(\mu, s) \notin \mathcal{H}$ **then**
$repeat \leftarrow \mathsf{false}$	$s \leftarrow\!\!\$\ \{0,1\}^\lambda$	$\mathcal{P}[\mu, s] \leftarrow \mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$
$(q, 2k, k, k) \leftarrow \mathsf{ParamSel}(1^\lambda, 3)$	**if** $(\mu, s) \in \mathcal{H}$	$\mathcal{H}[\mu, s] \leftarrow \mathfrak{h}_\psi(\mathcal{P}[\mu, s])$
$\psi \leftarrow\!\!\$\ \left(\mathbb{F}_q^{2k+1}\right)^\vee \otimes \left(\mathbb{F}_q^{k+1}\right)^\vee \otimes \left(\mathbb{F}_q^{k+1}\right)^\vee$	$repeat \leftarrow \mathsf{true}$	**return** $\mathcal{H}[\mu, s]$
$(\mu^\star, \widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}, s^*) \leftarrow \mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}(\psi)$	**abort**	
$b_0 := (\mu^\star, \cdot) \notin \mathcal{Q}$	$\mathsf{RO}(\mu, s)$	
$b_1 := \mathcal{H}[\mu^\star, s^\star] \stackrel{?}{=} \mathfrak{h}_\psi\left(\widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}\right)$	$\sigma \leftarrow (\mathcal{P}[\mu, s], s)$	
return $b_0 \wedge b_1$	$c \leftarrow c + 1$	
	$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mu, \sigma)\}$	
	return σ	

Fig. 10. Game 4. Trapdoor abandonment

the public tensor ψ uniformly from tensors of the same $(2k+1) \times (k+1) \times (k+1)$ format. By the pseudorandomness of non-singular Vandermone-Weyman-Zelevinsky tensor orbits, the distributions of Game_3 and Game_4 are computationally indistinguishable. In summary,

$$\left|\Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game}_0}(\psi) = 1\right] - \Pr\left[\mathcal{A}^{\mathsf{SIGN},\mathsf{RO}}_{\mathsf{Game}_4}(\psi) = 1\right]\right| \leq Q\ \mathsf{negl}(\lambda).$$

Now consider an adversary who wins Game_4 by returning a $(\mu^\star, \widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}, s^*)$ such that $\mu^\star$ was not queried for a signature and $\mathcal{H}[\mu^\star, s^\star] = \mathfrak{h}_\psi\left(\widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}\right)$. To do so, the adversary must know $\mathcal{H}[\mu^\star, s^\star]$, which is generated by the random oracle. Disregarding the possibility of the adversary luckily guessing this value, since it only happens with probability

$$\frac{1}{|\mathcal{S}_{k+1}(\mathbb{P}^{2k})|} \approx \frac{q}{\binom{2k+1}{k+1}(q-1)^{k+1}} << \frac{1}{2^\lambda},$$

we can conclude that adversary queried RO with the input $(\mu^\star, s^\star)$ to receive $\mathcal{P}[\mu, s]$, which in turn was drawn by $\mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$. But $\mathsf{SampleDom}^{\mathrm{3DB}}(\psi)$ draws $\mathcal{P}[\mu, s]$ uniformly from two preimages conditioned on the image $\mathfrak{h}_\psi(\mathcal{P}[\mu, s])$. Therefore, with probability at least a half, $\mathcal{P}[\mu, s] \neq \left(\widehat{w}^{\star(2)}, \widehat{w}^{\star(3)}\right)$. Therefore, an adversary who wins Game_4 finds a collision for the one way function h_ψ with probability at least a half. □

This Q query EUF-CMA$_Q$ in the ROM proof can also be lifted to the signature scheme Σ^{DB} in arbitrary dimensions. However, in the security analysis for Σ^{DB}, the domain sampler distribution conditioned on a target is not the same as that

of the preimage sampler. Therefore, the argument has to be modified with only the weaker guarantee that they are computationally indistinguishable.

5.2 Signature and key lengths

To meet the security parameter λ using the $(2k+1) \times (k+1) \times (k+1)$ format, it suffices to take $k = \lambda$ and $q \approx 4\lambda$. The signature length is dominated by the $\approx 2\lambda \log q$ bits to describe the projective vector pair $(\widehat{w}^{(2)}, \widehat{w}^{(3)})$ drawn by the preimage sampler. The other part of the signature is the salt, which takes λ bits to write down. As before, if we only consider polynomial time adversaries, which are bound to a polynomial number of queries in λ, we can get away with salts that are just super logarithmic $\omega(\log(\lambda))$ in length. Further, if an application bounds the number of signature queries corresponding to the same signing key, then the salt length can be lowered accordingly.

The public verification key length is nearly cubic in the security parameter λ since we have to explicitly write down the three dimensional tensor ψ. It gets longer exponentially in r, if we look to schemes in higher dimensions r. The four dimensional case might be an interesting compromise to study further, offering quartic length verification keys with an exponential support preimage to drawn from, for every target.

The secret signing key length is nearly quadratic in the security parameter λ, dominated by the need to write down the tuple of matrices describing the secret base change. This can be reduced to nearly linear in λ by considering pseudorandom invertible matrices described by seeds of length linear in λ. The trapdoor tensor matrix Λ only takes $\approx r\lambda \log q$ bits to describe, even in r dimensions.

But this signature size reduction strategy comes at the cost of increasing the computational complexity of the signing algorithm, which then has to expand the seeds to get the matrices and then invert them, paying the cost of the matrix multiplication exponent (or worse in practice) for the inversion. In particular, storing the inverses as part of the signing key to expedite the signing is ruled out in this strategy. A possible solution to having nearly linear length signing keys and nearly linear (or at least sub quadratic) time signing is to solve the following problem. Is there a way to generate a (pseudorandom matrix, its inverse) pair with both the matrix and its inverse having nearly linear length descriptions? Further, we want matrix-vector products with respect to both the matrix and its inverse to be efficient, say sub-quadratic. There are certainly structured matrices with these properties, such as Toeplitz or Cauchy or related family of matrices (or more generally matrices with displacement structure). The problem is to construct pseudorandom families with no visible structure. If one is prepared to make stronger hardness assumptions, one may even allow such structure, to make the private signing key and signature algorithm run time nearly linear in the security parameter.

5.3 Encryption

We next sketch an encryption scheme Π^{3DB} based on three dimensional doubly boundary $(2k+1) \times (k+1) \times (k+1)$ formats. Let the message space be b-bit strings. Fix a cryptographic hash function $\mathsf{H} : \{0,1\}^* \longrightarrow \{0,1\}^b$.

$\Pi^{\mathrm{3DB}}.\mathsf{KeyGen}(1^\lambda)$

$(q, 2k, k, k) \leftarrow \mathsf{ParamSel}(1^\lambda, 3)$
$(\psi, \mathsf{t}) \leftarrow \mathsf{TrapGen}\left(q, 1^{2k}, 1^k, 1^k\right)$
return $\mathsf{pk} = \psi, \mathsf{sk} = \mathsf{t}$

$\Pi^{\mathrm{3DB}}.\mathsf{Encrypt}(\mu, \mathsf{pk})$

$(\widehat{w}^{(2)}, \widehat{w}^{(3)}) \leftarrow \mathsf{SampleDom}^{\mathrm{3D}}(\psi)$
$c_1 \leftarrow h_\psi\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)$
$c_2 \leftarrow \mathsf{H}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \oplus \mu$
return (c_1, c_2)

$\Pi^{\mathrm{3DB}}.\mathsf{Decrypt}(\mathsf{sk}, c_1, c_2)$

$(\widehat{u}^{(2)}, \widehat{u}^{(3)}) \leftarrow \mathsf{SamplePre}(\mathsf{t}, c_1)$
$\mu' \leftarrow \mathsf{H}\left(\widehat{u}^{(2)}, \widehat{u}^{(3)}\right) \oplus c_2$
return μ'

Fig. 11. Encryption Π^{3DB} from 3D doubly boundary format tensors

The design is typical of encryption schemes built from trapdoor one way functions, but the choice of domain and preimage samplers we make is important. They have to be consistent, to avoid decryption failures. Our choice leads to perfect completeness, as we next argue. By lemma 4, for every target $\widehat{w}^{(1)} \in \mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$ on the Hamming sphere, $\mathsf{SamplePre}(\mathsf{t}, \widehat{w}^{(1)})$ draws a unique preimage that is the same under base change as the domain sample $\mathsf{SampleDom}^{\mathrm{3D}}(\psi)$ conditioned on its image being $\widehat{w}^{(1)}$. Therefore, in an honest execution of Π^{3DB}, $\left(\widehat{u}^{(2)}, \widehat{u}^{(3)}\right) = \left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)$ and

$$\mu' = \mathsf{H}\left(\widehat{u}^{(2)}, \widehat{u}^{(3)}\right) \oplus c_2 = \mathsf{H}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \oplus \mathsf{H}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \oplus \mu = \mu,$$

proving completeness of the protocol. The scheme as presented is $\mathsf{IND} - \mathsf{CPA}$ secure if we model the hash function as a random oracle, since the view of the adversary in the CPA game is identical, irrespective of which of the two possible challenge messages is chosen for encryption. In particular, say $(c_1, \mathsf{H}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \oplus \mu_x)$ is the encryption of the challenge message $\mu_x, x \leftarrow\$ \{0,1\}$. By lemma 3, the distribution of c_1 is uniform in the Hamming sphere $\mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right)$ and by design independent of the challenge message. The corresponding (image,preimage hash) pair $(c_1, \mathsf{H}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right))$ is indistinguishable from uniform in , unless the adversary can derive the preimage $\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)$ from c_1 and query the random oracle for

$\mathsf{H}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right)$. Since finding preimages is hard, $(c_1, \mathsf{H}\left(\widehat{w}^{(2)}, \widehat{w}^{(3)}\right) \oplus \mu_b)$ is indistinguishable from uniform in $\mathcal{S}_{k+1}\left(\mathbb{P}^{2k}\right) \times \{0,1\}^b$ and the scheme is CPA secure. Under the Fujisaki-Okomoto transformation [13], $\mathit{\Pi}^{\mathrm{3DB}}$ should give $\mathsf{IND} - \mathsf{CCA}_2$ secure encryption. We defer the formal security analysis of the encryption to a longer version of the paper.

References

1. Aguilar-Melchor, C., Blazy, O., Deneuville, J.C., Gaborit, P., Zémor, G.: Efficient encryption from random quasi-cyclic codes. IEEE Transactions on Information Theory **64**(5), 3927–3943 (2018). https://doi.org/10.1109/TIT.2018.2804444
2. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing. p. 99–108. STOC '96, Association for Computing Machinery, New York, NY, USA (1996). https://doi.org/10.1145/237814.237838, https://doi.org/10.1145/237814.237838
3. Ajtai, M.: Generating hard instances of the short basis problem. In: Wiedermann, J., van Emde Boas, P., Nielsen, M. (eds.) Automata, Languages and Programming. pp. 1–9. Springer Berlin Heidelberg, Berlin, Heidelberg (1999)
4. Alagic, G., Bros, M., Ciadoux, P., Cooper, D., Dang, Q., Thinh Dang, J.K., Lichtinger, J., Liu, Y.K., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., Silberg, H., Smith-Tone, D., Waller, N.: Status report on the first round of the additional digital signature schemes for the nist post-quantum cryptography standardization process. Tech. rep., NIST (2024), https://csrc.nist.gov/pubs/ir/8528/final
5. Alekhnovich, M.: More on average case vs approximation complexity. In: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science. p. 298. FOCS '03, IEEE Computer Society, USA (2003)
6. Bellare, M., Rogaway, P.: The exact security of digital signatures-how to sign with rsa and rabin. In: Maurer, U. (ed.) Advances in Cryptology — EUROCRYPT '96. pp. 399–416. Springer Berlin Heidelberg, Berlin, Heidelberg (1996)
7. Bläser, M., Duong, D.H., Narayanan, A.K., Plantard, T., Qiao, Y., Sipasseuth, A., Tang, G.: The alteq signature scheme: Algorithm specifications and supporting documentation (2023), https://pqcalteq.github.io/ALTEQ_spec_2023.09.18.pdf
8. Chen, Z., Grochow, J.A., Qiao, Y., Tang, G., Zhang, C.: On the Complexity of Isomorphism Problems for Tensors, Groups, and Polynomials III: Actions by Classical Groups. In: Guruswami, V. (ed.) 15th Innovations in Theoretical Computer Science Conference (ITCS 2024). Leibniz International Proceedings in Informatics (LIPIcs), vol. 287, pp. 31:1–31:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2024). https://doi.org/10.4230/LIPIcs.ITCS.2024.31, https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2024.31
9. Chou, T., Niederhagen, R., Persichetti, E., Randrianarisoa, T.H., Reijnders, K., Samardjiska, S., Trimoska, M.: Take your meds: Digital signatures from matrix code equivalence. In: El Mrabet, N., De Feo, L., Duquesne, S. (eds.) Progress in Cryptology - AFRICACRYPT 2023. pp. 28–52. Springer Nature Switzerland, Cham (2023)
10. Couvreur, A., Levrat, C.: Highway to hull: An algorithm for solving the general matrix code equivalence problem. In: Tauman Kalai, Y., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025. pp. 253–283. Springer Nature Switzerland, Cham (2025)

11. Emiris, I.Z., Mantzaflaris, A., Tsigaridas, E.: On the bit complexity of solving bilinear polynomial systems. In: Proceedings of the 2016 ACM International Symposium on Symbolic and Algebraic Computation. p. 215–222. ISSAC '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2930889.2930919, https://doi.org/10.1145/2930889.2930919
12. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) Advances in Cryptology — CRYPTO' 86. pp. 186–194. Springer Berlin Heidelberg, Berlin, Heidelberg (1987)
13. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology. p. 537–554. CRYPTO '99, Springer-Verlag, Berlin, Heidelberg (1999)
14. von zur Gathen, J., Gerhard, J.: Modern Computer Algebra. Cambridge University Press, 3 edn. (2013)
15. Gelfand, I., Kapranov, M., Zelevinsky, A.: Discriminants, Resultants, and Multidimensional Determinants. Modern Birkhäuser Classics, Birkhäuser Boston (2009), https://books.google.es/books?id=ZxeQBAAAQBAJ
16. Gelfand, I., Kapranov, M., Zelevinsky, A.: Hyperdeterminants. Advances in Mathematics **96**(2), 226–263 (1992). https://doi.org/https://doi.org/10.1016/0001-8708(92)90056-Q, https://www.sciencedirect.com/science/article/pii/000187089290056Q
17. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing. p. 197–206. STOC '08, Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1374376.1374407, https://doi.org/10.1145/1374376.1374407
18. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. J. ACM **38**(3), 690–728 (Jul 1991). https://doi.org/10.1145/116825.116852, https://doi.org/10.1145/116825.116852
19. Grochow, J., Qiao, Y.: On the complexity of isomorphism problems for tensors, groups, and polynomials i: Tensor isomorphism-completeness. SIAM Journal on Computing **52**(2), 568–617 (2023). https://doi.org/10.1137/21M1441110, https://doi.org/10.1137/21M1441110
20. Grochow, J.A., Qiao, Y.: On the complexity of isomorphism problems for tensors, groups, and polynomials iv: Linear-length reductions and their applications. In: Proceedings of the 57th Annual ACM Symposium on Theory of Computing. p. 766–776. STOC '25, Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3717823.3718282, https://doi.org/10.1145/3717823.3718282
21. Hillar, C.J., Lim, L.H.: Most tensor problems are np-hard. J. ACM **60**(6) (Nov 2013). https://doi.org/10.1145/2512329, https://doi.org/10.1145/2512329
22. Ji, Z., Qiao, Y., Song, F., Yun, A.: General linear group action on tensors: A candidate for post-quantum cryptography. In: Hofheinz, D., Rosen, A. (eds.) Theory of Cryptography. pp. 251–281. Springer International Publishing, Cham (2019)
23. Joux, A., Narayanan, A.K.: A High Dimensional Cramer's Rule Connecting Homogeneous Multilinear Equations to Hyperdeterminants. In: Meka, R. (ed.) 16th Innovations in Theoretical Computer Science Conference (ITCS 2025). Leibniz International Proceedings in Informatics (LIPIcs), vol. 325, pp. 62:1–62:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany

(2025). https://doi.org/10.4230/LIPIcs.ITCS.2025.62, https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2025.62
24. McEliece, R.J.: A Public-Key Cryptosystem Based On Algebraic Coding Theory. Deep Space Network Progress Report **44**, 114–116 (Jan 1978)
25. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology – EUROCRYPT 2012. pp. 700–718. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
26. Narayanan, A.K., Qiao, Y., Tang, G.: Algorithms for matrix code and alternating trilinear form equivalences via new isomorphism invariants. In: Advances in Cryptology – EUROCRYPT 2024: 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26–30, 2024, Proceedings, Part III. p. 160–187. Springer-Verlag, Berlin, Heidelberg (2024). https://doi.org/10.1007/978-3-031-58734-4_6, https://doi.org/10.1007/978-3-031-58734-4_6
27. Nguyen, P.Q., Regev, O.: Learning a parallelepiped: cryptanalysis of GGH and NTRU signatures. In: Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques. p. 271–288. EUROCRYPT'06, Springer-Verlag, Berlin, Heidelberg (2006). https://doi.org/10.1007/11761679_17, https://doi.org/10.1007/11761679_17
28. Patarin, J.: Hidden fields equations (hfe) and isomorphisms of polynomials (ip): Two new families of asymmetric algorithms. In: Maurer, U. (ed.) Advances in Cryptology — EUROCRYPT '96. pp. 33–48. Springer Berlin Heidelberg, Berlin, Heidelberg (1996)
29. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: Falcon. Tech. rep., Technical report, National Institute of Standards and Technology (2020), https://www.di.ens.fr/~prest/Publications/falcon.pdf
30. Rabin, M.O.: Digitalized signatures and public-key functions as intractable as factorization. Tech. rep., MIT, USA (1979)
31. Ran, L., Samardjiska, S.: Rare structures in tensor graphs - bermuda triangles for cryptosystems based on the tensor isomorphism problem. Cryptology ePrint Archive, Paper 2024/1396 (2024), https://eprint.iacr.org/2024/1396
32. Schulman, L.: Cryptography from tensor problems. IACR eprint (2012), https://eprint.iacr.org/2012/244
33. Spaenlehauer, P.J.: Solving multi-homogeneous and determinantal systems: algorithms, complexity, applications. Ph.D. thesis, Universitê Pierre et Marie Curie (Univ. Paris 6) (2012)
34. Spaenlehauer, P.J.: On the complexity of computing critical points with gröbner bases. SIAM Journal on Optimization **24**(3), 1382–1401 (2014). https://doi.org/10.1137/130936294, https://doi.org/10.1137/130936294
35. Sun, X.: Faster isomorphism for p-groups of class 2 and exponent p. In: Proceedings of the 55th Annual ACM Symposium on Theory of Computing. p. 433–440. STOC 2023, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3564246.3585250, https://doi.org/10.1145/3564246.3585250
36. Tang, G., Duong, D.H., Joux, A., Plantard, T., Qiao, Y., Susilo, W.: Practical post-quantum signature schemes from isomorphism problems of trilinear forms. Eurocrypt 2022 (2022), https://eprint.iacr.org/2022/267
37. Weyman, J., Zelevinsky, A.: Singularities of hyperdeterminants. Annales de l'Institut Fourier **46**(3), 591–644 (1996). https://doi.org/10.5802/aif.1526, http://www.numdam.org/articles/10.5802/aif.1526/