Finding isomorphisms between trilinear forms, slightly faster.

Anand Kumar Narayanan[1] Youming Qiao[2] Gang Tang[2,3]

[1]SandboxAQ, Palo Alto, United States.
anand.kumar@sandboxaq.com
[2]Centre for Quantum Software and Information, School of Computer Science, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW, Australia.
Youming.Qiao@uts.edu.au, gang.tang-1@student.uts.edu.au
[3] University of Birmingham, UK.

March 14, 2024

Given two square matrices A and B, can we tell if there is an invertible square matrix X such that XAX^{-1}=B? Yes, quickly! Such an X exists if and only if A and B are similar and there are efficiently computable invariants that precisely characterize similarity. Phrased differently, we can tell if there is a basis change that takes one matrix (or the corresponding bilinear form) to the other. But one small step in dimension, jumping from two (square matrices) to three (three dimensional tensors given by a cube of numbers), is a giant leap in computational complexity. Most linear algebraic problems concerning three dimensional tensors (or equivalently, trilinear forms) are (NP- or VNP- or #P-)hard. Two post-quantum signature submissions, **ALTEQ** and **MEDS**, are built on the hardness of finding a basis change that takes one of two given trilinear forms over finite fields to the other. **ALTEQ** and **MEDS** differ in the form of basis change under consideration. A basis change in **ALTEQ** corresponds to an action by an invertible matrix, while a triple of invertible matrices act in **MEDS**. Further, **ALTEQ** restricts the trilinear forms to be alternating.

In this talk, we present algorithms for solving the trilinear isomorphisms underlying **ALTEQ** and **MEDS** that improve upon previously known run time exponents by a constant factor. These algorithms inform the parameter selection in **ALTEQ** and **MEDS** and were already taken into account in the **ALTEQ** submission. Key ingredients in our algorithms are new distinguishing invariants under the respective actions. The run time analyses rely on certain heuristics, which are supported by experimental and theoretical evidence.

1 Introduction

Given two objects A and B of the same type, the *equivalence problem* asks if there exists a map π such that $\pi(A) = B$. The hardness of the equivalence problem depends on the objects and how the map is defined. There are objects in the equivalence problem that were recently proposed to support public-key cryptography for quantum-resistant purposes, such as linear or matrix codes [22,13,8], alternating trilinear form[42], lattice [26,25] etc.

Linear code equivalence. A classical equivalence problem is the *Code Equivalence* problem, which asks whether two given linear codes are isometric, that is, whether two linear codes are the same up to permuting, and possibly scalar multiplications on, the coordinates. One digital signature scheme submitted to the NIST call for additional signatures, LESS [5], is based on the assumed hardness of this problem.

Leon [36] initiated the study of this problem and proposed an algorithm that computes a list of both codes with minimum Hamming weight and then matches them to recover the isometry. Recently, Beullens [11] improved Leon's algorithm by using collision search. Another algorithm of significance is known as the Support Splitting Algorithm (SSA) by Sendrier [40]. Its running time increases exponentially in the dimension of the hull (the intersection of a code and its dual), and it works effectively for random linear codes under permutations. When scalar multiplications are also present, SSA works when $q \leq 4$ but not $q \geq 5$. If the hull is trivial and only permutations are used, then this problem can be reduced to graph isomorphism [7].

Matrix code equivalence. In this work, we are interested in the equivalence problem of matrix codes, called the *Matrix Code Equivalence* (MCE) problem. A matrix code over $\mathbb{F}_q$ is a linear subspace of the space of $m \times n$ matrices over $\mathbb{F}_q$. Concerning the MCE problem, it was recently shown to be at least as hard as the Code Equivalence problem [23,31], and to be equivalent to the homogeneous version of the Quadratic Maps Linear Equivalence (QMLE) problem [39,31].

Alternating trilinear form equivalence. We are also interested in another problem namely *Alternating Trilinear Form Equivalence* (ATFE), recently proposed in [42] to support a digital signature scheme. Here, the objects are alternating trilinear forms, namely a function $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ that is (1) linear in each argument, and (2) whenever two arguments are the same, ϕ evaluates to 0.

We now state the MCE and ATFE problems, which would also indicate what equivalences mean for matrix codes and alternating trilinear forms.

Definition 1 (Matrix Code Equivalence (MCE)). *Given two matrix codes $\mathcal{C}$ and $\mathcal{D}$ in $\mathrm{M}(m \times n, q)$, the problem asks whether there exist two invertible matrices $A \in \mathrm{GL}(m, q)$ and $B \in \mathrm{GL}(n, q)$ such that $\mathcal{D} = A\mathcal{C}B := \{ACB \mid C \in \mathcal{C}\}$.*

Definition 2 (Alternating Trilinear Form Equivalence (ATFE)). *Given two alternating trilinear forms $\phi, \psi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$, the problem asks whether there exists an invertible matrix $A \in \mathrm{GL}(n, q)$ such that for any $u, v, w \in \mathbb{F}_q^n$, $\phi(Au, Av, Aw) = \psi(u, v, w)$.*

MCE and ATFE: relations and cryptographic uses. MCE and ATFE are shown to be polynomial-time equivalent [32] and are Tensor Isomorphism (TI)-complete [31]. Utilising the MCE and ATFE problems, two signature schemes have recently been proposed by Tang et al. [42] and Chou et al. [22]. Both schemes are based on the Goldreich–Micali–Wigderson zero-knowledge protocol for graph isomorphism [30] and the Fiat–Shamir transformation [28]. More broadly these fall into the investigations on identifying and utilising group actions in cryptography [19,34,1]. These works lead to submissions to NIST's current standardization for post-quantum signatures: MEDS [21] and ALTEQ [15]. Subsequently, various applications have been developed, including ring signatures [14,24,22] and threshold signatures [9]. Hence, it is of significance to investigate the hardness of these two problems, as it will provide insights into the selection of secure parameter sets.

1.1 Previous works

In this section, we will briefly review some of state-of-the-art algorithms for MCE and ATFE. Algorithms for MCE and ATFE have been surveyed in [22] and [42], respectively. Beullens recently contributed beautiful new algorithms for ATFE in [12]. Here we explain two algorithms, one for MCE and one for ATFE, that are most relevant to us.

Leon-like algorithm for MCE. Leon's algorithm [36] is well-known for solving code equivalence problems in the Hamming metric. The key observation is that the equivalence preserves the Hamming weight of the codewords. Consequently, identifying the set of codewords with minimum Hamming weight within two codes can aid in revealing the equivalence or isometry between the codes. Recently, Beullens [11] improved upon this algorithm by constructing the set of codewords with a particular weight and the same multiset of entries as lists [4]. Subsequently, a collision search is conducted between the two lists to recover equivalence or isometry easily. It is natural to adapt Leon's algorithm to MCE [22]. That is, one can first build two lists of low-rank matrices in $\mathcal{C}_1$ and $\mathcal{C}_2$, and then do a collision search to find a matched pair of corresponding matrix codes and so recover the equivalence.

Beullens' algorithm for ATFE. Beullens [12] currently proposed a graph-theoretic algorithm to solve ATFE problem. An alternating trilinear form ϕ can be viewed

[4] In the monomial setting, Beullens considered building a set of 2-dimension subcodes with small support. This is because monomial transformation do not preserve anything beyond the hamming weight of a vector.

as a graph G_ϕ, where $\mathbf{v} \in \mathbb{F}_q^n$ is a vertex and $(\mathbf{u}, \mathbf{v})$ be an edge if and only if $\phi_{\mathbf{u},\mathbf{v}} = 0$. Also, a bilinear form $\phi_{\mathbf{u}}$ can be viewed as a matrix $M_{\phi,\mathbf{u}}$, then the rank of $\mathbf{u}$ is the rank of $M_{\phi,\mathbf{u}}$. The key observation is that the equivalence preserves the rank of the vertices in G_ϕ. Therefore, the algorithm first builds two lists of low-rank points in ϕ and ψ respectively and then finds a collision to recover the equivalence.

Gröbner basis approach. The MCE and ATFE problem can be solved algebraically by transforming them into a system of polynomial equations and then solving this system via Gröbner basis [42,22]. The Gröbner basis method, exhibits insensitivity to the parameter q within the system, with its efficiency contingent solely upon the values m, n and l (or n for the ATFE). Also, this approach demonstrates the high efficiency when applied to problems characterized by low dimensions.

1.2 Our contributions

In this paper, we propose heuristic algorithms for MCE and ATFE problems. We summarize our contributions as below.

Algorithm for MCE. We present a new algorithm for MCE. Our algorithm introduces a novel invariant for matrix codes, which we call the "corank-1 associated invariant". This innovation allows us to find a collision using the birthday paradox, and it avoids the use of Gröbner basis computations. This improvement leads to an algorithm with a complexity of $O(q^{(n-2)/2} \cdot (q \cdot n^3 + n^4) \cdot (\log(q))^2)$ as described in Section 4.4. We provide an implementation of this algorithm, and demonstrate its practical effectiveness for small n and q (such as $n = 9$ and $q = 31$) in Section 4.6.

Regarding the MEDS scheme, its security is based on the hardness of the MCE problem. Although our algorithm does not yet achieve a practical break of the parameter sets proposed by MEDS, it serves to underscore that these parameters have not yet attained the target security level; see Table 1.

parameter set	n	q	**Algebraic**	**Leon-like**	**Ours**
MEDS-I	14	4093	148.1	170.68	102.59
MEDS-III	22	4093	218.41	246.95	152.55
MEDS-V	30	2039	298.82	297.77	186.57

Table 1. Algorithms for solving the MCE problem. The data for algebraic and Leon-like algorithms are from the MEDS specification [21].

Importantly, we note that this could be fixed easily by enlarging q. This fix should not affect the running times, and only increase the signature sizes *at* *most*[5] linearly in $\log(q)$. Therefore the consequence of our algorithm on MEDS should be considered as mild.

Algorithm for ATFE. We present an algorithm for the ATFE problem by introducing a new isomorphism invariant. For an alternating trilinear form ϕ and a low-rank point v, the equivalence preserves the kernel space K of v. Based on this observation, we define an isomorphism invariant as the isomorphism type of the trilinear form $\hat{\phi}$ restricted to K in the first argument, under the action of $\mathrm{GL}(K) \times \mathrm{GL}(n, q)$. We provide preliminary evidence suggesting that this isomorphism invariant can be computed efficiently, and is distinguishing. *Assuming* that canonical forms for such restricted trilinear forms could also be computed efficiently, this leads to a birthday-type algorithm, with a complexity with the dominating factor being $O(q^{k/2})$, where $O(q^k)$ is the expected number of points with the target low rank. This could be compared with the algorithms in [12] with the dominating factors being $O(q^k)$ or $O(q^{n/2})$.

It must be noted that to utilise this invariant in a birthday-type algorithm, we need canonical forms rather than merely isomorphism testing. We were not able to derive such a canonical form algorithm, though we note that while to transform an isomorphism invariant algorithm to a canonical form may not be an easy process, it is generally regarded as doable, at least from the experience from graph isomorphism [3]. Therefore, protocol designers need to take the conservative approach, namely assuming a canonical form algorithm matching the isomorphism testing algorithm running time. This was the consideration when determining the parameters of ALTEQ [15].

Quantum speed-up. We accelerate our algorithms for both MCE and ATFE on quantum computers by using Szegedy's quantum random walks to find collisions [41]. The runtime exponent is reduced by a factor of 2/3, resulting in $q^{k/3}\mathsf{poly}(n, \log q)$ time quantum algorithms.

Our algorithms as a further development of [17,12]. Our algorithms for MCE and ATFE follow the previous works on polynomial isomorphism and alternating trilinear form equivalence. In particular, our algorithms are a further development of the works of Bouillaguet, Fouque, and Véber [17], and Beullens [12].

In [17], algorithms for testing isomorphism of systems of quadratic forms were presented. Both algorithms rely on certain graphs associated with quadratic form systems. The first algorithm in [17] samples a list of low-rank points for each of the two input polynomial systems, and find a collision which can be used in conjunction of the hybrid Gröbner basis method [27] to recover the secret transformation. The second algorithm in [17] works for $q = 2$; it is based on birthday paradox with an isomorphism invariant obtained by examining the radius-k neighbourhood of the points in the graph.

In [12], algorithms for ATFE were presented. Two of the algorithms that are most relevant to us are as follows. (We refer the reader to [12] for a beautiful algorithm for $n = 9$.) The first algorithm follows the sampling and collision approach,

[5] It is 'at most', because of the use of the seed tree techniques; see [22] for more details.

with the main innovation being that for the sampling step, where Beullens uses a random walk on the graph associated with an alternating trilinear form. The second algorithm is based on the birthday paradox with isomorphism invariants. As q is large for the use of ATFE in [42], Beullens used radius-1 or -2 neighbourhoods and observed that such neighbourhood information is distinguishing.

Our algorithms for MCE and ATFE are based on the birthday paradox with isomorphism invariants (see Section 3). As seen from the above, previous works use isomorphism invariants that are *local* (small radius neighbourhood) on graphs associated with polynomial systems or trilinear forms. Our main technical contribution is to discover new isomorphism invariants that can be viewed as transforming the information from graphs to *global* constraints.

For example, the isomorphism invariants for MCE are obtained by associating some graphs with matrix codes. We also perform a walk on the graph (starting from a corank-1 point), but we then use the path information to transform the matrix code as a whole to obtain an isomorphism invariant. Similarly, for ATFE, the isomorphism invariants are obtained by first taking the kernel of a low-rank point. We then apply this kernel to the alternating trilinear form to obtain another (smaller) trilinear form, and use this trilinear form as an isomorphism invariant.

Paper structure. After presenting preliminaries in Section 2, we present the generic algorithm framework we use in Section 3. We then describe the algorithm for MCE in Section 4, and the algorithm for ATFE in Section 5. Finally we present the quantum speed-ups for these algorithms in Section 6.

2 Preliminaries

Notations. For $n \in \mathbb{N}$, $[n] := \{1, 2, \ldots, n\}$. Let $\mathbb{F}_q$ be the finite field of q elements. We view $\mathbb{F}_q^n$ as the linear space of length-n column vectors over $\mathbb{F}_q$. Let $\mathbb{P} = \mathbb{P}(\mathbb{F}_q^n)$ be the projective space associated with the vector space $\mathbb{F}_q^n$. For a non-zero $\mathbf{u} \in \mathbb{F}_q^n$, we use $\hat{\mathbf{u}} \in \mathbb{P}$ to denote the projective line represented by $\mathbf{u}$. Let $\mathrm{GL}(n, q)$ denote the general linear group of degree n over $\mathbb{F}_q$. We use $\mathrm{M}(m \times n, q)$ to denote the space of $m \times n$ matrices over $\mathbb{F}_q$, and $\mathrm{ATF}(n, q)$ for the space of alternating trilinear forms over $\mathbb{F}_q^n$. For a finite set S, we use $s \leftarrow_R S$ to denote that s is uniformly randomly sampled from S.

Matrix codes and trilinear forms. A trilinear form is a function $\phi : \mathbb{F}_q^m \times \mathbb{F}_q^n \times \mathbb{F}_q^l \to \mathbb{F}_q$ that is linear in each of its three arguments.

Definition 3 (Trilinear Form Equivalence Problem). *Given two trilinear forms $\phi, \psi : \mathbb{F}_q^m \times \mathbb{F}_q^n \times \mathbb{F}_q^l \to \mathbb{F}_q$, the problem asks whether there exists three matrices $(A, B, C) \in \mathrm{GL}(m, q) \times \mathrm{GL}(n, q) \times \mathrm{GL}(l, q)$, such that for any $(u, v, w) \in \mathbb{F}_q^m \times \mathbb{F}_q^n \times \mathbb{F}_q^l$, $\phi(u, v, w) = \psi(A(u), B(v), C(w))$.*

A $[m \times n, l]$-matrix code $\mathcal{C}$ is an l-dimensional subspace of $\mathrm{M}(m \times n, q)$. We defined matrix code equivalence in Definition 1. Matrix code equivalence reduces to trilinear form equivalence in polynomial time. This is because of the following. Let a matrix code $\mathcal{C}$ be given by an ordered linear basis $(C_1, C_2, \dots, C_l)$, $C_k \in \mathrm{M}(m \times n, q)$, and $c_{i,j,k}$ denotes the (i, j)-entry of C_k. This gives rise to a trilinear form $\phi_{\mathcal{C}} : \mathbb{F}_q^m \times \mathbb{F}_q^n \times \mathbb{F}_q^l \to \mathbb{F}_q$, that is, $\phi_{\mathcal{C}} = \sum_{i,j,k} c_{i,j,k} u_i v_j w_k$ where $u = (u_1, \dots, u_m)^t \in \mathbb{F}_q^m$, $v = (v_1, \dots, v_n)^t \in \mathbb{F}_q^n$, and $w = (w_1, \dots, w_l)^t \in \mathbb{F}_q^l$. It is straightforward to verify that two matrix codes $\mathcal{C}$ and $\mathcal{D}$ are equivalent if and only if $\phi_{\mathcal{C}}$ and $\phi_{\mathcal{D}}$ are equivalent. Furthermore, if $(A, B, C) \in \mathrm{GL}(m, q) \times \mathrm{GL}(n, q) \times \mathrm{GL}(l, q)$ sends $\phi_{\mathcal{C}}$ ot $\phi_{\mathcal{D}}$, then (A, B) sends $\mathcal{C}$ to $\mathcal{D}$.

Alternating trilinear forms. A trilinear form $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ is alternating, if ϕ evaluates to 0 whenever two arguments are the same, e.g., $\phi(\mathbf{u}, \mathbf{u}, \mathbf{v}) = \phi(\mathbf{u}, \mathbf{v}, \mathbf{u}) = \phi(\mathbf{v}, \mathbf{u}, \mathbf{u}) = 0$ for all $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$. Let e_i be the ith standard basis vector and e_i^* be the corresponding dual basis which sends $\mathbf{u} = (u_1, \dots, u_n)^t \in \mathbb{F}_q^n$ to u_i. ϕ can be represented as $\sum_{1 \le i \le j \le k \le n} c_{i,j,k} e_i^* \wedge e_j^* \wedge e_k^*$ where $\wedge$ denotes the exterior product. And $e_i^* \wedge e_j^* \wedge e_k^*$ is an alternating trilinear form which can be defined as follows:

$$(e_i^* \wedge e_j^* \wedge e_k^*)(\mathbf{u}, \mathbf{v}, \mathbf{w}) = \det \begin{bmatrix} u_i & v_i & w_i \\ u_j & v_j & w_j \\ u_k & v_k & w_k \end{bmatrix},$$

where $\mathbf{u} = (u_1, \dots, u_n), \mathbf{v} = (v_1, \dots, v_n), \mathbf{w} = (w_1, \dots, w_n)$. This also implies that storing an alternating trilinear form requires $\binom{n}{3}$ field elements.

We note that the trilinear form equivalence problem differs from the alternating trilinear form equivalence problem, in that three invertible matrices are used in the former, while only one is used in the latter.

Instantiated arguments of trilinear forms. Let $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ be a trilinear form and $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$. We use $\phi(\mathbf{u}, \star, \star)$ to denote the bilinear form obtained by instantiating the first argument of ϕ with $\mathbf{u}$. Let $\phi(\mathbf{u}, \star, \star) = \sum_{j,k} c_{j,k} y_j z_k$ then it has matrix representation $M_{\mathbf{u}} = (c_{j,k})$ with respect to standard basis $e_1, \dots, e_n$. We use $\phi(\mathbf{u}, \mathbf{v}, \star)$ to denote the linear form obtained by instantiating the first two arguments of ϕ with $\mathbf{u}$ and $\mathbf{v}$, respectively.

Tripartite graphs associated with trilinear forms. Let $\phi \in \mathrm{TF}(\mathbb{F}_q^n)$ be a trilinear form, then we can associate ϕ with a tripartite graph $G_\phi = (U \uplus V \uplus W, E)$ where $U = V = W = \mathbb{P}(\mathbb{F}_q^n)$. To define the edge set E, let $\hat{\mathbf{u}} \in U$, $\hat{\mathbf{v}} \in V$, and $\hat{\mathbf{w}} \in W$. Then $\{\hat{\mathbf{u}}, \hat{\mathbf{v}}\} \in E$, if $\phi(\mathbf{u}, \mathbf{v}, \star)$ is the zero linear form. Similarly, $\{\hat{\mathbf{u}}, \hat{\mathbf{w}}\} \in E$, if $\phi(\mathbf{u}, \star, \mathbf{w})$ is the zero linear form. And $\{\hat{\mathbf{v}}, \hat{\mathbf{w}}\} \in E$, if $\phi(\star, \mathbf{v}, \mathbf{w})$ is the zero linear form.

Rank distribution of random trilinear forms. The following rank distribution of random trilinear forms follows from the well-known fact that the probability of a random matrix in $\mathrm{M}(n, \mathbb{F}_q)$ to be of rank $n - d$ tends to q^{-d^2} as $q \to \infty$ [10,29].

Theorem 1 ([10,29]). *Let n, d be positive integers such that $n - d$ is a non-negative number less than n. Then as $q \to \infty$, the average number of projective* *points with rank $n-d$ of a uniformly random trilinear form $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ tends to q^{-d^2+n-1}.*

Rank distribution of alternating trilinear forms. The following result is due to Beullens [12]; see also [14].

Theorem 2 ([12, Theorem 2]). *Let n, d be positive integers such that $n-d$ is a non-negative even number less than n. Then as $q \to \infty$, the average number of projective points with rank $n-d$ of a uniformly random alternating trilinear form $\phi \in \mathrm{ATF}(\mathbb{F}_q^n)$ tends to $q^{(-d^2+3d)/2+n-2}$.*

3 Finding equivalences of trilinear forms via invariants

We first outline the common framework of our algorithms for ATFE and TFE at a high level, following Beullens (in Section 5.4 of [12]). But in a departure from [12] which relies on invariants derived from graphs on projective points, we design new global invariants. The invariant functions for ATF and TF will be of the form

$$F_0 : \mathrm{TF}(\mathbb{F}_q^n) \times \mathbb{P}(\mathbb{F}_q^n) \to X_0,$$
$$F_1 : \mathrm{ATF}(\mathbb{F}_q^n) \times \mathbb{P}(\mathbb{F}_q^n) \to X_1$$

and explicitly constructed in the following sections. The subscript 0 in the function and the target set indicates that it is associated with TF. Likewise, the subscript 1 indicates association with ATF.

Invariants. To illustrate the notion of invariants, let us first name the actions underlying MCE and ATFE in the language of trilinear forms.

Definition 4 (MCE Action). *For a trilinear form $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \longrightarrow \mathbb{F}_q$ and a triple of matrices $(A, B, C) \in \mathrm{GL}(n, q)^3$, define the trilinear form*

$$\begin{aligned} \phi_{A,B,C} : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n &\longrightarrow \mathbb{F}_q \\ (x, y, z) &\longmapsto \phi(Ax, By, Cz). \end{aligned}$$

We design F_0 as a pairing of the trilinear form and the projective space that is invariant under twisting the trilinear form and the projective space. The trilinear form is twisted by the $\mathrm{GL}(n, q)^3$ MCE Action. The projective space is twisted by the inverse of the matrix acting on the first dimension of the trilinear form. Formally, the invariant for MCE action needs to satisfy that

$$\forall \phi \in \mathrm{TF}(\mathbb{F}_q^n), \forall \hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n), \forall (A, B, C) \in \mathrm{GL}(n, q)^3, F_0(\phi, \hat{\mathbf{v}}) = F_0(\phi_{A,B,C}, A^{-1}\hat{\mathbf{v}}).$$

Definition 5 (ATFE Action). *For a trilinear form $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \longrightarrow \mathbb{F}_q$ and a matrix $A \in \mathrm{GL}(n, q)$, define the trilinear form*

$$\begin{aligned} \phi_A : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n &\longrightarrow \mathbb{F}_q \\ (x, y, z) &\longmapsto \phi(Ax, Ay, Az). \end{aligned}$$

We design the function F_1 as a pairing of the trilinear form and the projective space that is invariant under twisting the trilinear form by the ATFE action and the projective space by the inverse of the matrix defining the ATFE action. Formally,

$$\forall \phi \in \mathrm{ATF}(\mathbb{F}_q^n), \forall \hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n), \forall A \in \mathrm{GL}(n,q), F_1(\phi, \hat{\mathbf{v}}) = F_1(\phi_A, A^{-1}\hat{\mathbf{v}}).$$

Distinguishing invariant. The invariant function F_0 is called distinguishing if for all $\phi \in \mathrm{TF}(\mathbb{F}_q^n)$,

$$\Pr_{(\hat{\mathbf{v}}_1,\hat{\mathbf{v}}_2)\leftarrow_R \mathbb{P}(\mathbb{F}_q^n)^2} (F_0(\phi,\hat{\mathbf{v}}_1) \neq F_0(\phi,\hat{\mathbf{v}}_2)) \approx 1.$$

We will specify the meaning of ≈ 1 in the following. Likewise, F_1 is called distinguishing if for all $\phi \in \mathrm{ATF}(\mathbb{F}_q^n)$,

$$\Pr_{(\hat{\mathbf{v}}_1,\hat{\mathbf{v}}_2)\leftarrow_R \mathbb{P}(\mathbb{F}_q^n)^2} (F_1(\phi,\hat{\mathbf{v}}_1) \neq F_1(\phi,\hat{\mathbf{v}}_2)) \approx 1.$$

An algorithm template based on distinguishing invariants. With such distinguishing invariant functions at hand, we have the following generic algorithm for MCE and ATFE. The version for ATFE is specified in parentheses.

To start with, recall that for a trilinear form $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ and $\mathbf{v} \in \mathbb{F}_q^n$, the rank of $\phi(\mathbf{v},\star,\star)$ (see Section 2) is an invariant, which has been utilised in [17,12]. Also note that $\mathrm{rk}(\phi(\mathbf{v},\star,\star)) = \mathrm{rk}(\phi(\lambda\mathbf{v},\star,\star))$ for non-zero $\lambda \in \mathbb{F}_q$, so we can talk about the rank of $\phi(\hat{\mathbf{v}},\star,\star)$ for $\hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n)$.

This rank invariant cannot be distinguished. Still, the new invariants considered in this paper are further refinements of the rank invariant, as will be seen below. In particular, the generic algorithm is parametrised by this rank R, which would be specified later depending on the specific invariants.

Input: Two equivalent (alternating) trilinear forms $\phi, \psi \in \mathrm{TF}(\mathbb{F}_q^n)$(or $\mathrm{ATF}(\mathbb{F}_q^n)$).
Output: $A, B, C \in \mathrm{GL}(n,q)$ such that $\phi_{A,B,C} = \psi$ (or $A \in \mathrm{GL}(n,q)$ such that $\phi_A = \psi$).
Algorithm
1. Pick a positive number $R \leq n$. Let

$$\mathbb{P}_{\phi,R} := \{\hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n) \mid \mathrm{rk}(\phi(\hat{\mathbf{v}},*,*)) = R \ ,$$

$$\mathbb{P}_{\psi,R} := \{\hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n) \mid \mathrm{rk}(\psi(\hat{\mathbf{v}},*,*)) = R$$

denote the respective set of points where the trilinear forms specialize in the first dimension to give rank R matrices. Independently sample a set $L_{\phi,R}$ of $\sqrt{|\mathbb{P}_{\phi,R}|}$ points from $\mathbb{P}_{\phi,R}$ and a set $L_{\psi,R}$ of $\sqrt{|\mathbb{P}_{\psi,R}|}$ points from $\mathbb{P}_{\psi,R}$. Since ϕ and ψ are isomorphic, $\mathbb{P}_{\phi,R} = \mathbb{P}_{\psi,R}$ and we denote their cardinality as $N_R := \|\mathbb{P}_{\phi,R}\| = \|\mathbb{P}_{\psi,R}\|$. Therefore $L_{\phi,R}$ and $L_{\psi,R}$ are both $\sqrt{N_R}$-sized subsets of the same set of size N_R.
2. Apply the invariant function F_i (where $i = 0$ for MCE and $i = 1$ for ATFE) to each element in $L_{\phi,R}$ and $L_{\psi,R}$. Find a pair $(\hat{\mathbf{v}}, \hat{\mathbf{v}'})$ for which $F_i(\phi,\hat{\mathbf{v}}) = F_i(\psi,\hat{\mathbf{v}'})$, where $\hat{\mathbf{v}} \in L_{\phi,R}$ and $\hat{\mathbf{v}'} \in L_{\psi,R}$. The existence of such a pair is likely due to the birthday paradox.

3. For MCE, such a pair reveals the desired output $(A, B, C) \in \mathrm{GL}(n, q)^3$ through linear algebra, as we describe in Section 4. To solve the ATFE, feed the matching pair $(\hat{\mathbf{v}}, \hat{\mathbf{v}'})$ as the partial information into the Gröbner basis computation in [42,6]. This Gröbner basis computation is a heuristic that finds in polynomial time an $A \in \mathrm{GL}(n, q)$ (if it exists) such that $\phi_A = \psi$ and $A^{-1}\hat{\mathbf{v}} = \hat{\mathbf{v}'}$.

The complexity of the above algorithm parameterized by the target rank R can be estimated as

$$O\left(\sqrt{N_R} \cdot (\mathsf{samp\text{-}cost} + \mathsf{inv\text{-}cost}) + \mathsf{recover\text{-}cost}\right). \tag{1}$$

The sampling cost samp-cost refers to the cost of sampling a rank-R (projective) point, that is, a point in $\mathbb{P}_{\phi,R}$ (or equivalently in $\mathbb{P}_{\psi,R}$). And inv-cost denotes the cost of invariant computation for each point. The cost of recovering the isomorphism given a collision is denoted by recover-cost. Also note that for the invariant to be distinguishing enough in the above procedure, we need to have $\Pr_{(\hat{\mathbf{v}}_1,\hat{\mathbf{v}}_2)\leftarrow_R \mathbb{P}(\mathbb{F}_q^n)^2}\left(F_0(\phi,\hat{\mathbf{v}}_1) = F_0(\phi,\hat{\mathbf{v}}_2)\right) = O(1/N_R)$.

In the following two sections, we describe algorithms in this general framework tailored to MCE and ATFE, by describing the invariant functions and optimizing the rank R.

4 An algorithm for Matrix Code Equivalence

In this section, we introduce an algorithm for the matrix code (or trilinear form) equivalence problem. Specifically, given two trilinear forms $\phi \in \mathrm{TF}(\mathbb{F}_q^n)$ and $\psi \in \mathrm{TF}(\mathbb{F}_q^n)$ that are equivalent, the algorithm computes an equivalence $(A, B, C) \in \mathrm{GL}(n, q) \times \mathrm{GL}(n, q) \times \mathrm{GL}(n, q)$ between ϕ and ψ. The algorithm runs in time $O(q^{(n-2)/2} \cdot (q \cdot n^3 + n^4) \cdot (\log(q))^2)$.

4.1 The main idea

To instantiate the algorithm outlined in Section 3, the primary bottleneck is identifying invariants with sufficient distinguishing power. The main idea of the algorithm is to associate distinguishing invariants to corank-1 points, specifically for those $\hat{\mathbf{u}} \in \mathbb{P}(\mathbb{F}_q^n)$ such that the bilinear form $\phi(\mathbf{u}, \star, \star)$ is of rank $n-1$. We shall occasionally call such projective lines as corank-1 points. Recall there is a tripartite graph $G_\phi = (U \uplus V \uplus W, E)$ associated with ϕ where $U = V = W = \mathbb{P}(\mathbb{F}_q^n)$. Each corank-1 point $\hat{\mathbf{u}} \in U$ has a unique neighbour $\hat{\mathbf{v}} \in V$, namely the one dimensional left kernel of the bilinear form $\phi(\mathbf{u}, \star, \star)$. Since $\phi(\star, \mathbf{v}, \star)$ has $\mathbf{u}$ in its left kernel, $\phi(\star, \mathbf{v}, \star)$ has co-rank at least 1. If $\phi(\star, \mathbf{v}, \star)$ is of corank-1, it has a unique neighbour $\hat{\mathbf{w}} \in W$. Repeating this procedure leads to a path on G_ϕ. We continue building this path until reaching length $3n$, collecting n points each from U, V and W. Such a path is built without ambiguity if and only if at every iteration we get a point of corank-1.

Our experiments show that for most starting points $\hat{\mathbf{u}}$, we do obtain a path of length $3n$ without ambiguity and that the vector n-tuples collected in each of the sets U, V and W are linearly independent respectively. We use these three vector tuples to transform ϕ to $\tilde{\phi}[\mathbf{u}]$ which depends only on the vectors on this path.

To make this an isomorphism invariant indexed with $\hat{\mathbf{u}}$ (instead of with $\mathbf{u}$), we need to remove the ambiguity caused by the scalar multiples, which can be done easily by locating non-zero evaluations of $\tilde{\phi}[\mathbf{u}]$ on about $3n$ inputs of the form (e_i, e_j, e_k). This gives us $\bar{\phi}[\hat{\mathbf{u}}]$ which is an invariant associated with $\hat{\mathbf{u}}$. Our experiments show that this invariant is distinguishing, i.e. different $\hat{\mathbf{u}}$ results in different $\bar{\phi}[\hat{\mathbf{u}}]$. This allows for an application of the birthday algorithm.

It is known from Theorem 1 that for a random ϕ, there exist approximately q^{n-2} corank-1 points. Thus we get an algorithm running in time $O((q^{(n/2)} + q^{(n-2)/2}) \cdot \mathsf{poly}(n, q))$ by instantiating the above invariant.

4.2 From a vector to three vector tuples

Corank-1 points of trilinear forms and paths on G_ϕ. Suppose a non-zero $\mathbf{u}_1 \in \mathbb{F}_q^n$ satisfies that $\phi(\mathbf{u}_1, \star, \star)$ is of corank-1 as a bilinear form. Consider the following steps.

1. As $\phi(\mathbf{u}_1, \star, \star)$ is of corank-1, there exists a unique $\hat{\mathbf{v}_1} \in \mathbb{P}$ such that $\phi(\mathbf{u}_1, \mathbf{v}_1, \star)$ is the zero linear form.
2. If $\phi(\star, \mathbf{v}_1, \star)$ is of corank-1, then there exists a unique $\hat{\mathbf{w}_1} \in \mathbb{P}$, such that $\phi(\star, \mathbf{v}_1, \mathbf{w}_1)$ is the zero linear form.
3. If $\phi(\star, \star, \mathbf{w}_1)$ is of corank-1, then there exists a unique $\hat{\mathbf{u}_2} \in \mathbb{P}$, such that $\phi(\mathbf{u}_2, \star, \mathbf{w}_1)$ is the zero linear form.

If $\hat{\mathbf{u}_1} \neq \hat{\mathbf{u}_2}$, then the above procedure produces a path $(\hat{\mathbf{u}_1}, \hat{\mathbf{v}_1}, \hat{\mathbf{w}_1}, \hat{\mathbf{u}_2})$ in $G(\phi)$. We can continue the above procedure as follows.

1. Let $L_U = (u_1)$, $L_V = ()$, and $L_W = ()$.
2. For $i = 1$ to n, do the following:
   (a) Compute the unique $\hat{\mathbf{v}}_i \in \mathbb{P}(\mathbb{F}_q^n)$, such that $\phi(\mathbf{u}_i, \mathbf{v}_i, \star) = 0$.
   (b) If the corank of $\phi(\star, \mathbf{v}_i, \star)$ is not 1, or if $\mathbf{v}_i \in \mathrm{span}(L_V)$, terminate and report "Fail". Otherwise, add $\mathbf{v}_i$ to L_V.
   (c) Compute the unique $\hat{\mathbf{w}}_i \in \mathbb{P}(\mathbb{F}_q^n)$, such that $\phi(\star, \mathbf{v}_i, \mathbf{w}_i) = 0$.
   (d) If the corank of $\phi(\star, \star, \mathbf{w}_i)$ is not 1, or if $\mathbf{w}_i \in \mathrm{span}(L_W)$, terminate and report "Fail". Otherwise, add $\mathbf{w}_i$ to L_W.
   (e) If $i = n$, break.
   (f) Compute the unique $\hat{\mathbf{u}_{i+1}} \in \mathbb{P}(\mathbb{F}_q^n)$, such that $\phi(\mathbf{u}_{i+1}, \star, \mathbf{w}_i) = 0$.
   (g) If the corank of $\phi(\mathbf{u}_{i+1}, \star, \star)$ is not 1, or if $\mathbf{u}_{i+1} \in \mathrm{span}(L_U)$, terminate and report "Fail". Otherwise, add $\mathbf{u}_{i+1}$ to L_U.

If the above procedure does not return "Fail", then we obtain three vector tuples $L_U = (\mathbf{u}_1, \ldots, \mathbf{u}_n)$, $L_V = (\mathbf{v}_1, \ldots, \mathbf{v}_n)$, and $L_W = (\mathbf{w}_1, \ldots, \mathbf{w}_n)$, such that $\mathbf{u}_i$'s (resp, $\mathbf{v}_i$'s, $\mathbf{w}_i$'s) are linearly independent.

4.3 Corank-1 invariants from three vector tuples

Suppose that starting from a corank-1 $\mathbf{u}_1 \in \mathbb{F}_q^n$, we obtain three vector tuples L_U, L_V, and L_W, which are canonically associated with $\mathbf{u}_1$. We then treat L_U, L_V, and L_W as invertible matrices, that is, $L_U = \begin{bmatrix}\mathbf{u}_1 \ldots \mathbf{u}_n\end{bmatrix}^t$. Define a trilinear form $\tilde{\phi} : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ by $\tilde{\phi}(x, y, z) = \phi(L_U(x), L_V(y), L_W(z))$. This $\tilde{\phi}$ is almost an isomorphism invariant associated with $\mathbf{u}_1$ – almost because there is an ambiguity associated with the representing vectors of $\hat{\mathbf{u}}_i$, $\hat{\mathbf{v}}_j$, and $\hat{\mathbf{w}}_k$.

To remove this ambiguity, we need to study the canonical form of $\tilde{\phi}$ under the action of $\mathrm{D}(n, q) \times \mathrm{D}(n, q) \times \mathrm{D}(n, q)$, where $\mathrm{D}(n, q)$ denotes the group of invertible diagonal $n\times n$ matrices over $\mathbb{F}_q$. This can be done by carefully selecting $3n$ non-zero entries in $\tilde{\phi}$, so that the diagonal entries of the acting matrices are determined by these entries. In the following we present one choice of non-zero entries. There could be several other natural selections depending on the positions of zero entries, but we do not pursue them further as this choice is already useful enough in our practical implementation.

Consider the following entries. For any $i, j, k \geq 3$,

$$\begin{aligned}&a_i := \tilde{\phi}(e_i, e_2, e_1), b_j := \tilde{\phi}(e_1, e_j, e_1), c_k := \tilde{\phi}(e_1, e_2, e_k), d_1 := \tilde{\phi}(e_1, e_2, e_1),\\ &d_2 := \tilde{\phi}(e_2, e_3, e_5), d_3 := \tilde{\phi}(e_1, e_3, e_2), d_4 := \tilde{\phi}(e_2, e_1, e_2) \text{ are non-zero.}\end{aligned} \tag{2}$$

In this case, we can use the action of $\mathrm{D}(n, q) \times \mathrm{D}(n, q) \times \mathrm{D}(n, q)$ to set a_i, b_j, c_k, d_1, d_2, d_3 and d_4 to be 1. More specifically, let $(F, G, H) \in \mathrm{D}(n, q) \times \mathrm{D}(n, q) \times \mathrm{D}(n, q)$, where $F = \mathrm{diag}(f_1, \ldots, f_n)$, $G = \mathrm{diag}(g_1, \ldots, g_n)$, and $H = \mathrm{diag}(h_1, \ldots, h_n)$. Then set f_i, g_j, and h_k to satisfy that, for $3 \leq i, j, k \leq n$,

$$\begin{aligned}&f_1g_2h_1 = 1/d_1, f_i/f_1 = d_1/a_i, g_j/g_2 = d_1/b_j, h_k/h_1 = d_1/c_k,\\ &f_2 = 1/(g_3h_5d_2), h_2 = 1/(f_1g_3d_3), g_1 = 1/(f_2h_2d_4).\end{aligned} \tag{3}$$

Let $\bar{\phi} : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ be defined by $\bar{\phi}(x, y, z) = \tilde{\phi}(F(x), G(y), H(z))$. Then $\bar{\phi}(e_i, e_j, e_k) = f_ig_jh_k\tilde{\phi}(e_i, e_j, e_k)$. Therefore,

$$\bar{\phi}(e_1, e_2, e_1) = f_1g_2h_1\tilde{\phi}(e_1, e_2, e_1) = 1/d_1 \cdot d_1 = 1.$$

For $i \geq 3$,

$$\begin{aligned}&\bar{\phi}(e_i, e_2, e_1)\\ &= f_ig_2h_1\tilde{\phi}(e_i, e_2, e_1)\\ &= (f_i/f_1)f_1g_2h_1\tilde{\phi}(e_i, e_2, e_1)\\ &= (d_1/a_i) \cdot (1/d_1) \cdot a_i = 1.\end{aligned}$$

Similarly, it can be verified that $\bar{\phi}(e_1, e_j, e_1) = \bar{\phi}(e_1, e_2, e_k) = 1$ for $j, k \geq 3$. Additionally, we can verify that $\bar{\phi}(e_1, e_2, e_1) = \bar{\phi}(e_2, e_3, e_5) = \bar{\phi}(e_2, e_1, e_2) =$

$\bar{\phi}(e_1, e_3, e_2) = 1$. Furthermore, for any $i, j, k \geq 3$,

$$\begin{aligned} &\bar{\phi}(e_i, e_j, e_k) \\ &= f_i g_j h_k \tilde{\phi}(e_i, e_j, e_k) \\ &= (f_i/f_1)(g_j/g_2)(h_k/h_1) f_1 g_2 h_1 \tilde{\phi}(e_i, e_j, e_k) \\ &= \frac{d_1^4 \tilde{\phi}(e_i, e_j, e_k)}{a_i b_j c_k}; \end{aligned}$$

for $i = 2$ and any $j, k \geq 3$,

$$\begin{aligned} &\bar{\phi}(e_2, e_j, e_k) \\ &= f_2 g_j h_k \tilde{\phi}(e_2, e_j, e_k) \\ &= (f_2/f_1)(g_j/g_2)(h_k/h_1) f_1 g_2 h_1 \tilde{\phi}(e_2, e_j, e_k) \\ &= \frac{b_3 b_5 \tilde{\phi}(e_2, e_j, e_k)}{d_1 d_2 b_j c_k}; \end{aligned}$$

for $k = 2$ and any $i, j \geq 3$,

$$\begin{aligned} &\bar{\phi}(e_i, e_j, e_2) \\ &= f_i g_j h_2 \tilde{\phi}(e_i, e_j, e_2) \\ &= (f_i/f_1)(g_j/g_2)(h_2/h_1) f_1 g_2 h_1 \tilde{\phi}(e_i, e_j, e_2) \\ &= \frac{b_3 \tilde{\phi}(e_i, e_j, e_2)}{d_3 a_i b_j}; \end{aligned}$$

for $j = 1$ and any $i, k \geq 3$,

$$\begin{aligned} &\bar{\phi}(e_i, e_1, e_k) \\ &= f_i g_1 h_k \tilde{\phi}(e_i, e_1, e_k) \\ &= (f_i/f_1)(g_1/g_2)(h_k/h_1) f_1 g_2 h_1 \tilde{\phi}(e_i, e_1, e_k) \\ &= \frac{d_1^7 d_2 d_3 d_4 \tilde{\phi}(e_i, e_1, e_k)}{b_3^2 b_5 a_i c_k}; \end{aligned}$$

So $\bar{\phi}$ is completely determined by the conditions in Equation 3.

The above suggests that $\bar{\phi}[\hat{\mathbf{u}}_1] := \bar{\phi}$ is an isomorphism invariant associated with $\hat{\mathbf{u}}_1 \in \mathbb{P}(\mathbb{F}_q^n)$, assuming that $\tilde{\phi}$ satisfies Equation 2.

4.4 Description of the algorithm

Given the above preparations, the algorithm works as follows.

Input. Two equivalent trilinear forms $\phi, \psi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$.
Output. An equivalence $(A, B, C) \in \mathrm{GL}(n, q) \times \mathrm{GL}(n, q) \times \mathrm{GL}(n, q)$.
Algorithm. 1. For ϕ, construct a list S_ϕ of $q^{(n-2)/2}$ corank-1 $\hat{\mathbf{u}} \in \mathbb{P}$ together with the isomorphism invariant $\bar{\phi}[\hat{\mathbf{u}}]$ as follows.

   (a) Compute one corank-1 $\hat{\mathbf{u}} \in \mathbb{P}$ by sampling randomly $\mathbf{u} \in \mathbb{F}_q^n$ q times.
   (b) For $\hat{\mathbf{u}} \in \mathbb{P}$, compute three vector tuples L_U, L_V, and L_W as in Section 4.2.
   (c) Use L_U, L_V and L_W to transform ϕ to $\tilde{\phi}[\mathbf{u}]$.
   (d) Use the method in Section 4.3 to transform $\tilde{\phi}[\mathbf{u}]$ to $\bar{\phi}[\hat{\mathbf{u}}]$.
2. For ψ, construct a list S_ψ of $q^{(n-2)/2}$ corank-1 $\hat{\mathbf{u}} \in \mathbb{P}(\mathbb{F}_q^n)$ together with the isomorphism invariant $\bar{\psi}[\hat{\mathbf{u}}]$ as above.
3. Find $\hat{\mathbf{u}}$ from S_ϕ, and $\hat{\mathbf{u}'}$ from S_ψ, such that $\bar{\phi}[\hat{\mathbf{u}}]$ and $\bar{\psi}[\hat{\mathbf{u}'}]$ are the same.
4. An equivalence (A, B, C) from ϕ to ψ can be obtained by composing the transformations from ϕ to $\bar{\phi}[\hat{\mathbf{u}}]$ and from ψ to $\bar{\psi}[\hat{\mathbf{u}'}]$.

Time analysis of the above algorithm. We assume that the modular arithmetic complexity in $\mathbb{F}_q$ is in time $O((\log q)^2)$, and the number of arithmetic operations for $n \times n$ matrix computations (such as matrix multiplication and rank computation) is $O(n^3)$. As in the practical setting, n is small and matrices are dense, this should be a reasonable estimate (rather than using $O(n^\omega)$ where ω is the matrix multiplication exponent).

Step 1 is a For-loop contributing a multiplicative factor of $q^{(n-2)/2}$ to steps (a) to (d). Step (a) samples vectors in $\mathbb{F}_q^n$ and computes the ranks of the associated matrices for q times, so its complexity is $O(q \cdot (n \cdot \log(q) + n^3 \cdot (\log q)^2))$. Step (b) constructs three n-tuples of vectors. Each vector in this n-tuple is obtained by solving a system of n linear equations in n variables. So Step (b) costs $O(n \cdot n^3 \cdot (\log q)^2) = O(n^4 \cdot (\log q)^2)$. Step (c) requires $3n$ $n \times n$ matrix multiplications, so its complexity is also $O(n^4 \cdot (\log q)^2)$. For Step (d), the method in Section 4.3 takes $O(n^3 \cdot (\log q)^2)$ time. Taking into account of the For-loop factor, the total cost for steps 1 and (a) to (d) is $O(q^{(n-2)/2} \cdot (q \cdot n^3 + n^4) \cdot (\log(q))^2)$.

Once the two lists are constructed, finding a collision and using that to construct an isomorphism takes time $O(\log(q^{(n-2)/2}))$ as we can assume that the lists S_ϕ and S_ψ are sorted. Therefore steps 2 to 4 contribute to a running time of lower order, and the running time of the whole algorithm is $O(q^{(n-2)/2} \cdot (q \cdot n^3 + n^4) \cdot (\log(q))^2)$.

Correctness analysis of the above algorithm. We assume that $\bar{\phi}[\hat{\mathbf{u}}]$ is a distinguishing invariant of $\hat{\mathbf{u}}$. Then by birthday paradox, the above algorithm returns $\hat{\mathbf{u}}$ from S_ϕ, and $\hat{\mathbf{u}'}$ from S_ψ, such that $\bar{\phi}[\hat{\mathbf{u}}]$ and $\bar{\psi}[\hat{\mathbf{u}'}]$ are the same, with constant probability.

4.5 Heuristic assumptions for the invariant

We now reflect on several assumptions required for using $\bar{\phi}[\mathbf{u}_1]$ for $\mathbf{u}_1 \in \mathbb{F}_q^n$ with $\phi(\mathbf{u}_1, \star, \star)$ being of corank-1.

1. We assume that we can obtain three vector tuples L_U, L_V, L_W.
2. We assume that $\tilde{\phi}$, the trilinear form obtained after applying L_U, L_V, and L_W, satisfies Equation 2.
3. We assume that the corank-1 invariant $\bar{\phi}[\mathbf{u}_1]$ is distinguishing.

We next argue in favour of each of these heuristics.

Heuristic 1. To build the vector tuples L_U, L_V, and L_W, it suffices (1) to perform a walk with corank-1 points for $3n$ successful steps, and (2) the vectors in L_U (resp. L_V, L_W) be linearly independent.

We argue for (1), by making the same assumption as in Beullens' algorithms [12], namely those points along such a walk are close to independent randomly sampled. In particular, the probability of getting a walk with corank-1 points for $3n$ steps can be estimated as follows. The probability of a corank-1 point having a corank-2 neighbour is asymptotically $O(1/q^2)$; this can be calculated following the techniques in [12]. Therefore, the probability of walking for $3n$ steps with corank-1 points is lower bounded by $1 - O(n/q^2)$, assuming points along such a walk are close to independent randomly sampled.

We argue for (2) using algebraic-geometry. To this end, consider a generic starting corank-1 vector $\mathbf{u}_1$ and think of its coordinate vector $(\mathbf{u}_{1,1}, \mathbf{u}_{1,2}, \ldots, \mathbf{u}_{1,n})$ as n indeterminates. The corank-1 assumption implies that there is a unique projective $\hat{\mathbf{v}}_1$ such that $\phi(\mathbf{u}_1, \mathbf{v}_1, *) = 0$ (that is, the zero dual vector). The coordinates of $\mathbf{v}_1$ can be expressed as some vector of polynomials in the coordinate ring of $\mathbf{u}_1$, for instance using the adjugate matrix of $\phi(\mathbf{u}_1, *, *)$. Call this vector of polynomials as $(f^{\phi}_{\mathbf{v}_1,j})_{1\leq j\leq n} \in (\mathbb{F}_q[\mathbf{u}_{1,1}, \mathbf{u}_{1,2}, \ldots, \mathbf{u}_{1,n}])^n$. The superscript ϕ signifies that the coefficients of each $f^{\phi}_{\mathbf{v}_1,j}$ depend only on the tensor ϕ. Repeating a similar process starting with the coordinate vector $(f^{\phi}_{\mathbf{v}_1,j})_{1\leq j\leq n}$ of $\mathbf{v}_1$, we obtain the coordinates $(f^{\phi}_{\mathbf{w}_1,j})_{1\leq j\leq n} \in (\mathbb{F}_q[\mathbf{u}_{1,1}, \mathbf{u}_{1,2}, \ldots, \mathbf{u}_{1,n}])^n$ of $\mathbf{w}_1 \in L_W$. Note that each coordinate is a polynomial in the coordinate ring of the generic starting vector $\mathbf{u}_1$. Continuing this way, we can express each element of L_U, L_V, and L_W as a vector of polynomials in the co-ordinate ring of $\mathbf{u}_1$. The vectors in L_U being linearly independent can be expressed as a polynomial condition on the coordinates of $\mathbf{u}_1$, namely the determinant of the matrix $(f^{\phi}_{\mathbf{u},j})_{u\in L_U, 1\leq j\leq n}$ vanishing. In particular, the variety of dependent L_U has co-dimension at least one, as long as the symbolic determinant $\det\left((f^{\phi}_{\mathbf{u},j})_{u\in L_U,1\leq j\leq n}\right)$ is not identically zero. The matrix $(f^{\phi}_{\mathbf{u},j})_{u\in L_U,1\leq j\leq n}$ depends only on ϕ. For the random choice of ϕ induced by key generation, the symbolic determinant $\det\left((f^{\phi}_{\mathbf{u},j})_{u\in L_U,1\leq j\leq n}\right)$ is almost certainly not identically zero. Therefore, its roots, which constitutes the pathological variety of dependent L_U has co-dimension at least one. Therefore with probability at least $1 - 1/q$, we expect the co-ordinates of a random starting vector $\mathbf{u}_1$ to not be in this variety, implying that the L_U vectors are linearly independent. The probability $1-1/q$ is only a crude estimate. For a precise bound taking into account the structure of the polynomial, we can invoke the Schwartz–Zippel lemma or more generally the Lang–Weil bound. The Lang–Weil bound subsumes the Schwartz–Zippel lemma and gives stronger bounds in many cases where more (such as number of irreducible components, degree, smoothness, etc.) is known about the polynomial $\det\left((f^{\phi}_{\mathbf{u},j})_{u\in L_U,1\leq j\leq n}\right)$. In either case, to unconditionally prove that a random $\mathbf{u}_1$ is not in this variety, it helps if the degree of the polynomial is not too big. Naively, the polynomial produced through expansion is of exponential degree, but this is unlikely to be optimal, as shown in the experiment part. We leave an unconditional proof of the validity of this heuristic to future work.

Heuristic 2. Here we assume that $O(n)$ entries in the transformed tensor are non-zero. Therefore, the probability of this assumption failing increases as q decreases and n increases. Note that this assumption is used only to deal with diagonal group actions, and more specialized techniques can be done to reduce the failure probability of this step.

Heuristic 3. We prove that the invariants generated by our algorithm are distinguishing with high probability, under the following well studied conjecture from [39], which we re-phrase in tensor notation. To this end, define the automorphism group of a tensor $\phi \in TF(\mathbb{F}_q)$ as the subgroup $\mathrm{Aut}(\phi) \leqslant \mathrm{GL}(n,q)^3$ such that

$$\forall (A,B,C) \in \mathrm{Aut}(\phi), \forall (x,y,z) \in \mathbb{F}_q^n, \phi(Ax,By,Cz) = \phi(x,y,z).$$

Clearly, scalar matrix triples of the form

$$\{(\lambda I_n, \mu I_n, \nu I_n) \mid \lambda\mu\nu = 1, (\lambda,\mu,\nu) \in \left(\mathbb{F}_q^\times\right)^3\} \leqslant \mathrm{Aut}(\phi)$$

form a subgroup of the automorphism group. We say that the automorphism group $\mathrm{Aut}(\phi)$ is trivial or equivalently that ϕ has trivial automorphism group if and only if

$$\{(\lambda I_n, \mu I_n, \nu I_n) \mid \lambda\mu\nu = 1, (\lambda,\mu,\nu) \in \left(\mathbb{F}_q^\times\right)^3\} = \mathrm{Aut}(\phi).$$

That is, all automorphisms are merely triples of scalar matrices.

Conjecture 1. For uniformly random $\phi \in TF(\mathbb{F}_q^n)$, with probability negligibly close to 1, the automorphism group $\mathrm{Aut}(\phi)$ is trivial.

This conjecture is stated as a "mild assumption" in [39], where the authors provide convincing theoretic and empirical evidence. In fact, this conjecture is assumed true in half of the complexity theoretic reductions in the web of problems centered around MCE ([39, Fig. 1]), that lay as the foundation for MEDS.

Consider the corank-1 invariant $\bar{\phi}[\hat{\mathbf{u}}]$ constructed at a successful completion of the first step of the algorithm. We prove in the subsequent Lemma 1 that $\bar{\phi}[\hat{\mathbf{u}}]$ is distinguishing if the isomorphism class of ϕ has a trivial automorphism group.

Lemma 1. *If $\phi \in TF(\mathbb{F}_q^n)$ has the trivial automorphism group, then the isomorphism invariant $(\phi, \hat{\mathbf{u}}) \longmapsto \bar{\phi}[\hat{\mathbf{u}}]$ determined by step 1 of the algorithm is distinguishing.*

Proof. Recall the notation in the description of the algorithm, to aid in the proof sketch. Let (L_U, L_V, L_W) and (L'_U, L'_V, L'_W) be the two vector tuples produced starting from different $\mathbf{u}$ and $\mathbf{u}'$, respectively. Let $\bar{\phi}[\hat{\mathbf{u}}]$ and $\bar{\phi}[\hat{\mathbf{u}}]$ respectively denote the images of the invariant computed by step 1 of the algorithm. If the algorithm samples two $\bar{\phi}[\hat{\mathbf{u}}]$ and $\bar{\phi}[\hat{\mathbf{u}}]$ that are the same, then the respective vector tuples (L_U, L_V, L_W) and (L'_U, L'_V, L'_W) can be composed to get a non-trivial automorphism in $\mathrm{Aut}(\phi)$. But $\phi \in TF(\mathbb{F}_q^n)$ has the trivial automorphism group, therefore $\bar{\phi}[\hat{\mathbf{u}}]$ and $\bar{\phi}[\hat{\mathbf{u}}]$ are distinct, implying the invariant is distinguishing.

The MEDS key generation algorithm chooses a ϕ uniformly at random from $TF(\mathbb{F}_q^n)$. Assuming conjecture 1, $\mathrm{Aut}(\phi)$ is trivial with probability negligibly close to 1. Therefore, lemma 1 applies in our setting (except possibly with negligibly small probability), implying $(\phi, \bar{\mathbf{u}}) \longmapsto \bar{\phi}[\hat{\mathbf{u}}]$ is distinguishing.

Experimental support. We carry out experiments on Magma [16] for $n = 6$ to 10 and $q = 1021$ to verify the assumptions as above.

We examine Assumptions 1, 2, and 3 sequentially as follows. That is, for a point $\mathbf{u}$, we first verify if assumption 1 holds. If so, then we check if assumption 2 holds for $\mathbf{u}$. If both assumptions 1 and 2 hold, we call $\mathbf{u}$ an *effective point*. In Table 2, we sample 1000 points, and record the number of points failing assumption 1, and the number of points satisfying assumption 1 but failing assumption 2, as well as the number of effective points.

Finally, to verify assumption 3, we do experiments on these effective points. Our results show that for the instances in the Table 2, the isomorphism invariants corresponding to all points are pairwise distinguishable. This is expected, because each sample is generated randomly, these points are essentially distinct from one another.

$q \backslash n$	6	7	8	9	10	11	12	13	14
509	7/26/967	1/39/960	5/40/955	5/41/954	1/70/929	12/58/930	6/57/937	11/67/922	5/81/914
1021	8/10/982	5/16/979	10/20/970	4/28/968	2/18/980	1/27/972	3/31/966	2/30/968	1/29/970
2039	1/13/986	1/13/986	3/14/983	2/8/990	0/18/982	0/18/982	1/15/984	2/17/981	0/18/982
4093	1/5/994	1/7/992	1/5/994	1/7/992	0/6/994	2/6/992	0/13/987	2/11/987	0/10/990
8191	0/3/997	0/2/998	1/2/997	0/2/998	1/4/995	0/3/997	0/5/995	1/8/991	1/5/994
16381	0/0/1000	0/1/999	0/4/996	0/0/1000	0/4/996	0/1/999	0/3/997	1/4/995	0/3/997

$q \backslash n$	15	16	17	18	19	20	21	22
509	1/88/911	11/99/890	6/90/904	3/119/878	3/104/893	7/99/894	6/128/866	3/116/881
1021	1/27/972	3/45/952	5/49/946	1/54/945	5/58/937	2/54/944	2/67/931	7/59/934
2039	4/18/978	1/19/980	0/28/972	2/20/978	2/25/973	2/31/967	2/29/969	2/28/970
4093	2/8/990	1/10/989	1/18/981	0/16/984	3/15/982	1/23/976	1/11/988	1/22/977
8191	1/3/996	0/4/996	1/7/992	0/4/996	1/10/989	1/9/990	0/4/996	0/8/992
16381	0/7/993	0/2/998	0/1/999	0/1/999	0/8/992	0/4/996	0/3/997	1/3/996

Table 2. Statistics of effective points. a/b/c in the table are defined as follows: a (resp. b) is the number of points for which Assumption 1 (resp. Assumption 2) does not hold, and c is the number of effective points.

Note that it is enough for all but a small fraction of corank-1 $\mathbf{u}_1$ to satisfy the above. Furthermore, if some assumption is not satisfied, this would also constitute as an invariant. That is, if $\mathbf{u}_1, \ldots, \mathbf{u}_i$ in L_U becomes linearly dependent, then this number i also becomes an invariant which can be utilised. We do not attempt to deal with such cases because they rarely happen in experiments.

4.6 Experimental results for the algorithm

We implemented the algorithm in Section 4.4 in Magma [16]. We tested our implementation on a server (AMD EPYC 7532 CPU at 2.40GHz) to solve some instances of the MCE problem. The results are given in Table 3. Our experiments demonstrate that when running ten instances, two to four of them successfully discover collisions and recover the secret matrices (A, B, C).

Because we conduct $q^{(n-2)/2}$ samplings, we cannot set q to be too large for a practical running. Therefore, we set q to be 61 or 31. As a result, the fraction of effective points is not as large as for $q = 1021$ as in Table 2. For example, in MCE-instance-1, we conducted 3721 samplings and obtained 2702 effective points. Therefore, when q is large, the success rate should increase with the number of effective points.

Parameter set	n	q	Number of effective points	Number of sampling times	Time (seconds)
MCE-instance-1	6	61	2702	3721	420
MCE-instance-2	7	61	20053	29062	5638
MCE-instance-3	8	61	149149	226981	100900
MCE-instance-4	9	31	64202	165870	137715

Table 3. Solving MCE instances

Remark 1. Following [12], a possible improvement on the sampling step (Step (a) of the algorithm in Section 4.4) is as follows.

Recall that in Step (a) of the algorithm in Section 4.4, a corank-1 point is obtained by sampling a random vector in $\mathbb{F}_q^n$ for q times. However, note that starting from a corank-1 vector $\hat{\mathbf{u}}$, the vectors in the vector tuple L_U, if successfully built, are all corank-1. So these vectors can be utilised, instead of starting from a fresh random corank-1 vector. In general, we can walk along the path in the tripartite graph starting from a corank-1 vector until we hit a vector of corank larger than 1. This has the potential of reducing the complexity of the algorithm from $O(q^{(n-2)/2} \cdot (q \cdot n^3 + n^4) \cdot (\log(q))^2)$ to $O(q^{(n-2)/2} \cdot n^4 \cdot (\log(q))^2)$, as we would only need to sample a fresh corank-1 vector very few times during the execution of the algorithm.

One question for this approach is whether it results in a distribution close to the uniform one. To test this, we implemented the above approach. In the case of MCE-instance-1, our preliminary experimental results show that when running 6 instances, one of them successfully finds a collision and recovers the secret matrices. We leave a more careful analysis and more experiments to a future work.

5 An algorithm for alternating trilinear form equivalence

In this section, we present our algorithm for the ATFE problem. That is, given two alternating trilinear forms $\phi \in \mathrm{ATF}(\mathbb{F}_q^n)$ and $\psi \in \mathrm{ATF}(\mathbb{F}_q^n)$, the algorithm computes an equivalence $A \in \mathrm{GL}(n, q)$ from ϕ to ψ, if such A exists.

As will be explained later, there is a component missing for implementing this algorithm for ATFE, namely the transformation of isomorphism testing procedures to canonical forms. (On the contrary, the corresponding component in our algorithm for matrix code equivalence is automatically a canonical form algorithm.) Still, as it is usually the case that an isomorphism testing algorithm can be turned into a canonical form algorithm (such as for graph isomorphism [4]), the time complexity of this algorithm is used in the parameter setup of ALTEQ [15].

Before introducing our algorithm, we review the algorithms for ATFE by Beullens [12], which inspire our algorithm.

5.1 Beullens' algorithms for ATFE

In [12], Beullens presented some novel algorithms for ATFE. Here we describe two algorithms there that work for general n.

The first algorithm is a collision algorithm based on low-rank points based on the graph-walking sampling method. That is, suppose a random $\phi \in \mathrm{ATF}(n, q)$ has approximately q^k-many projective points of rank r. Then for $\phi, \psi \in \mathrm{ATF}(n, q)$ that are equivalent via $A \in \mathrm{GL}(n, q)$, one can sample $q^{1/2 \cdot k}$-many rank-r points for ϕ, and another $q^{1/2 \cdot k}$-many rank-r points for ψ. Then by the birthday paradox, with constant probability there exists a pair of points $(\mathbf{u}, \mathbf{v})$ from these two lists, such that $A(\mathbf{u}) = \mathbf{v}$. Combined with a Gröbner basis with partial information procedure[6], this correspondence enables to recover the whole A. To sample rank-r points, Beullens invented the graph-walk sampling method, which allows for sampling e.g. corank-3 points for odd n more efficiently than directly using min-rank for relatively small q. The major cost of this approach is usually the collision step, with time complexity $q^k \cdot \mathrm{poly}(n, \log q)$.

The second algorithm is a birthday algorithm based on isomorphism invariants. Such an algorithm was already proposed for the polynomial isomorphism problem by Bouillaguet, Fouque, and Véber in [17] for $q = 2$. Beullens observed that for radius-1 or 2 neighbours of corank-1 (for odd n) or corank-2 (for even n), the rank information should serve as a distinguishing isomorphism invariants. The major cost of this approach is the number of corank-1 or corank-2 points, so Beullens estimated the running time as $q^{n/2+c} \cdot \mathrm{poly}(n, \log q)$.

[6] Beullens discovered that Gröbner basis with partial information still works well given (1) a correspondence between projective points, and (2) the kernel information of low-rank points.

5.2 An algorithm for ATFE based on a new isomorphism invariant

The main innovation of our algorithm for ATFE is to associate distinguishing isomorphism invariants to low-rank points.

Let $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$. Suppose by Theorem 2, it is expected that there are roughly q^k many $\hat{\mathbf{u}} \in \mathbb{P}(\mathbb{F}_q^n)$, such that $\mathrm{rk}_\phi(\hat{\mathbf{u}}) = r$. Let us *assume* that there is an easy-to-compute, distinguishing, isomorphism invariant[7] for those rank-r $\hat{\mathbf{u}}$.

Then the algorithm goes as follows: first sample $O(q^{k/2})$-many rank-r points for ϕ, and $O(q^{k/2})$-many rank-r points for ψ. For each point, compute this isomorphism invariant. Then by the birthday paradox, there exist one point $\hat{\mathbf{u}}$ from the list of ϕ, and one point $\hat{\mathbf{v}}$ from the list of ψ, such that their isomorphism invariants are the same. Finally, use Gröbner basis with partial information for $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ to recover the desired isomorphism.

Following Equation 1, the running time of the above algorithm can then be estimated as

$$O(q^{k/2} \cdot (\mathsf{samp\text{-}cost} + \mathsf{inv\text{-}cost}) + \mathsf{gb\text{-}cost}),$$

where samp-cost denotes the sampling cost, the inv-cost denotes the invariant computing cost, and gb-cost denotes the Gröbner basis with partial information cost.

The sampling step can be achieved by either the min-rank method (Appendix A) or Beullens' graph-walking method [12]. For the min-rank method, the cost of sampling a low-rank matrix can be estimated for concrete values of n, k, and r by e.g. [6,35,44]. For the graph-walking method, the sampling cost can be estimated based on certain statistics of graphs associated with alternating trilinear forms by Beullens [12, Theorem 1].

The gb-cost can be estimated as $O(n^6)$ as in [12]. This is based on the hybrid Gröbner basis method with the first row known in the variable matrix. The effectiveness of this hybrid Gröbner basis method was first discovered in [27] and then utilised in [17,42]. Beullens further improved this method by noting that (1) knowing the first row up to scalar suffices, and (2) for low-rank points, the kernel information can be incorporated [12, Section 4].

The main innovation of the above algorithm is a new isomorphism invariant which we describe next.

5.3 The isomorphism invariant step

Suppose $\hat{\mathbf{u}} \in \mathbb{P}(\mathbb{F}_q^n)$ satisfies that $\mathrm{rk}_\phi(\hat{\mathbf{u}}) = r$. Then $K := \ker(\phi_{\hat{\mathbf{u}}}) \leq \mathbb{F}_q^n$ is a dimension-$(n-r)$ space, also preserved by any isomorphism. This allows us to consider the trilinear form $\tilde{\phi}_{\hat{\mathbf{u}}} : K \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$, and it can be verified easily that the *isomorphism type* of $\tilde{\phi}_{\hat{\mathbf{u}}}$ under $\mathrm{GL}(K) \times \mathrm{GL}(n, q)$ is an isomorphism invariant.

[7] That is a function f from low-rank points to some set S, such that $f(\hat{\mathbf{u}}) \neq f(\hat{\mathbf{v}})$ for $\hat{\mathbf{u}} \neq \hat{\mathbf{v}}$, and f is unchanged by basis changes.

To use the isomorphism type of $\tilde{\phi}_{\hat{\mathbf{u}}}$ in the algorithm, we need the isomorphism types are (1) easy to compute, and (2) distinguishing; that is, for different $\hat{\mathbf{u}}, \hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n)$, $\tilde{\phi}_{\hat{\mathbf{u}}}$ and $\tilde{\phi}_{\hat{\mathbf{v}}}$ are different.

To verify these, we perform the following experiment in Magma [16].

1. Sample a random $\phi \in \mathrm{ATF}(n, q)$.
2. Sample a random rank-r point $\hat{\mathbf{u}} \in \mathbb{P}(\mathbb{F}_q^n)$.
3. Sample t random rank-r points $\hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n)$. For each such point, do:
   (a) Use the Gröbner basis with partial information to decide whether $\tilde{\phi}_{\hat{\mathbf{u}}}$ and $\tilde{\phi}_{\hat{\mathbf{v}}}$ are isomorphic.

Our experiments give the following.

- For $n = 9$, $r = 4$, and $p = 3$, 10 experiments (i.e. for 10 $\hat{\mathbf{u}}$ from 10 random alternating trilinear forms) with $t = 100$ comparisons (i.e. for 100 different $\hat{\mathbf{v}}$ to compare with $\hat{\mathbf{u}}$). On average, 75 out of 100 $\tilde{\phi}_{\hat{\mathbf{v}}}$ are not isomorphic with $\tilde{\phi}_{\hat{\mathbf{u}}}$.
- For $n = 10$, $r = 6$, and $p = 3$, 10 experiments (i.e. for 10 $\hat{\mathbf{u}}$ from 10 random alternating trilinear forms) with $t = 100$ comparisons (i.e. for 100 different $\hat{\mathbf{v}}$ to compare with $\hat{\mathbf{u}}$). On average, 95 out of 100 $\tilde{\phi}_{\hat{\mathbf{v}}}$ are not isomorphic with $\tilde{\phi}_{\hat{\mathbf{u}}}$.

For $n = 11$, our code does not work for $n = 11$ on a laptop, due to the Gröbner basis step.

From these experiments we see that (1) the Gröbner basis with partial information algorithm is effective in practice to test isomorphism between $\tilde{\phi}_{\hat{\mathbf{u}}}$ and $\tilde{\phi}_{\hat{\mathbf{v}}}$, and (2) as n goes from 9 to 10, the isomorphism type of $\tilde{\phi}_{\hat{\mathbf{u}}}$ becomes more distinguishing. These give some preliminary support that the isomorphism types of $\tilde{\phi}_{\hat{\mathbf{u}}}$ do serve as a easy-to-compute, distinguishing, isomorphism invariant.

Note that testing isomorphism here is not enough, and canonical forms are required to serve as an isomorphism invariant. Even though to transform an isomorphism invariant algorithm to a canonical form one may not be an easy process, it is generally regarded as doable, at least from the experience from graph isomorphism [3].

5.4 Concrete estimations of this algorithm for ALTEQ parameters

We show the improvement of our algorithm over Beullens' algorithm for a set of ALTEQ parameters. In [15], $n = 13$ and $q = 2^{32} - 1$ are used for the 128-bit security. In this case, Beullens' algorithm runs in time $O(q^{(n-5)/2} \cdot n^{11} + q^{n-7} \cdot n^6)$. As the major factor comes from q^{n-7}, the bit complexity is above $32 \cdot 6 = 192$. For our algorithm, using rank-$(n-5)$ points, the time complexity is estimated as $O(q^{(n-7)/2} \cdot (\mathsf{samp\text{-}cost} + \mathsf{inv\text{-}cost}) + \mathsf{gb\text{-}cost})$. The sampling cost can be estimated as in Appendix A based on [6], which is 32-bit complexity. The inv-cost and gb-cost are lower than the sampling cost. So the total bit complexity of our algorithm is $32 \cdot 3 + 32 = 128$.

6 Quantum attacks

We lower the run time exponent of our classical algorithms for MCE and ATFE on a quantum computer by a factor of 2/3. This speed up results from deploying Szegedy type quantum random walks to find collisions, but comes at the cost of exponential quantum space requirement. Therefore, there is reason to only consider the classical algorithms to tune the parameters of the cryptosystems. We describe the quantum algorithms for ATFE in greater detail. The MCE case is analogous but a little easier, since there is no need for Gröbner basis computations.

6.1 Collision detection through quantum random walks

The first collision detection quantum algorithms were due to Brassard, Høyer, and Tapp [18] and special to two-to-one functions, building on Grover's search [33]. Ambanis removed these restrictions and devised improved collision detection algorithms through quantum random walks, that match lower bounds [2]. Szegedy further improved these algorithms and brought them under a unified framework of quantum random walks with memory [41]. We will use Szegedy's version of quantum random walks for the quantum speedups of classical algorithms to the decision version ATFE.

We first paraphrase theorem 3 in [41], specialized to the oracle function being the identity. Let X be a finite set and $R \subset X \times X$ a binary relation with a membership tester. For a positive real number α and a uniformly random subset $H \subset X$ of size $|X|^\alpha$, let p_α denote the probability that $R \cap (H \times H)$ is non empty. There is a quantum algorithm to differentiate between the cases $p_\alpha = 0$ and $p_\alpha \geq \epsilon$ in time $O(|X|^\alpha + 1000\sqrt{|X|^\alpha/\epsilon})$.

Extensions of Szegedy's algorithm by Magniez, Nayak, Richter, Roland, and Santha [38,37] may be deployed to tackle the search version ATFE within the same running time. Another extension of Szegedy's algorithm is to claw finding, by Tani [43]. The claw finding formalism is convenient to phrase ATFE in and infer polynomial speed ups. Let $f : X \to Z$ and $g : Y \to Z$ be two functions between finite sets. Given oracle access to f and g, the claw finding problem is to find an $(x, y) \in X \times Y$ such that $f(x) = g(y)$, if one exists. The functions may be presented either as standard oracles or as comparison oracles. We describe the later in the quantum setting, as they suffice. A comparison oracle maps quantum states

$$|x, y, b, w\rangle \longmapsto |x, y, b \oplus [f(x) >^? g(y)], w\rangle.$$

Here, b is a bit; x and y respectively index quantum states corresponding to elements in X and Y. Fixing an ordering on Z, $[f(x) >^? g(y)]$ is a bit that is one if and only if $f(x) > g(y)$. The last register indexed by w is an ancilla for work space. For instances with X and Y of roughly the same size, Tani's algorithm finds claws on a quantum computer in time $O((|X||Y|)^{1/3})$.

In applying these quantum random walk algorithms, we will invoke generic algorithms applicable to functions on finite sets presented as an oracle. For clarity of exposition, we focus on speedups to the main exponential term and suppress incremental polynomial factors.

6.2 Solving ATFE through quantum random walks.

As a warm up, we first describe quantum algorithms for ATFE that do not exploit our new invariants. Then, we build on these algorithms by incorporating the invariants to achieve the aforementioned run time exponent.

A classical oracle from the Gröbner basis attack with partial information. First, consider the decision version of ATFE. That is, given two alternating trilinear forms ϕ and ψ, the existence of an $A \in GL(n,q)$ such that $\psi = \phi \circ A$ is in question. Central to all our methods is a polynomial time classical algorithm to test membership in the relation set

$$R_{\phi,\psi} := \{(\hat{\mathbf{u}}, \hat{\mathbf{v}}) \in \mathbb{P}(\mathbb{F}_q^n)^2 \mid \exists A \in GL(n,q) \text{ such that } \psi = \phi \circ A \text{ and } A^{-1}\hat{\mathbf{u}} = \hat{\mathbf{v}}\}.$$

If ϕ and ψ are not isomorphic, $R_{\phi,\psi}$ is empty. A pair $(\hat{\mathbf{u}}, \hat{\mathbf{v}}) \in \mathbb{P}(\mathbb{F}_q^n)^2$ satisfying $A^{-1}\hat{\mathbf{u}} = \hat{\mathbf{v}}$ enforces n $\mathbb{F}_q$-linear constraints on A. The Gröbner basis attack with partial information in [27], augmented with these linear constraints can tell in heuristic polynomial time if the pair $(\hat{\mathbf{u}}, \hat{\mathbf{v}})$ is in $R_{\phi,\psi}$. We henceforth make the same assumptions. This polynomial time classical algorithm to test membership can be converted to a polynomial sized quantum circuit that can test membership in superposition. Further, incorporate a time out clause into the membership algorithm to make the Gröbner basis methods stop searching and declare non existence.

Invoke Szegedy's algorithm with X as $\mathbb{P}(\mathbb{F}_q^n)$, R as $R_{\phi,\psi}$, α as $1/3$ and uniformly sampling an $H \subset \mathbb{P}(\mathbb{F}_q^n)$ of size $\Theta\left(q^{n/3}\right)$. We claim that the probability gap may be taken to be $\epsilon = \Omega\left(q^{-n/3}\right)$. To prove the claim, consider two isomorphic ϕ and ψ. That is, there exists at least one $A_{\phi,\psi} \in \mathrm{GL}(n,q)$ such that $\psi = \phi \circ A_{\phi,\psi}$. Therefore,

$$\Pr_H\left((R_{\phi,\psi} \cap (H \times H)) \neq \emptyset\right) \geq \Pr_H\left((H \cap A_{\phi,\psi}(H)) \neq \emptyset\right) \geq \Omega\left(q^{-n/3}\right),$$

proving the claim. In summary, we can tell if ϕ and ψ are isomorphic in time $q^{n/3}\mathrm{poly}(n, \log q)$ on a quantum computer. This strategy also tackles the promise search version ATFE within the same running time, thanks to extensions of Szegedy's algorithm by Magniez, Nayak, Richter, Roland, and Santha [38,37]. An alternative is to solve ATFE by claw finding. To phrase ATFE as claw finding, independently draw uniformly random sets $X \subset \mathbb{P}(\mathbb{F}_q^n)$ and $Y \subset \mathbb{P}(\mathbb{F}_q^n)$, each of size $q^{n/2}$. Take $f : X \to \mathbb{P}(\mathbb{F}_q^n)$ as the multiplication by A^{-1} map $\mathbf{u} \longmapsto A^{-1}\mathbf{u}$ and $g : Y \to \mathbb{F}_q^n$ as the identity. The birthday paradox ensures for isomorphic ϕ and ψ that there is a solution to claw finding with constant positive probability. The algorithm for testing membership in $R_{\phi,\psi}$ from the previous subsection yields a comparison oracle. Tani's algorithm for claw finding solves ATFE in time $q^{n/3}\mathrm{poly}(n, \log q)$.

6.3 Low-rank birthday attacks on ATFE via quantum random walks

We next describe how our invariant functions can be incorporated into the quantum algorithms. For $\phi \in \mathrm{ATF}(\mathbb{F}_q^n)$ and $\hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n)$, let $\phi/\hat{\mathbf{v}}$ denote the isomorphism class of the restriction of ϕ to $\mathsf{ker}(\phi_{\hat{\mathbf{v}}}) \times \mathbb{F}_q^n \times \mathbb{F}_q^n$ under the $\mathrm{GL}(\mathsf{ker}(\phi_{\hat{\mathbf{v}}})) \times \mathrm{GL}(n,q)$ action. For a positive number R, let

$$S_R := \left\{(\phi, \hat{\mathbf{v}}) \in \mathrm{ATF}(\mathbb{F}_q^n) \times \mathbb{P}(\mathbb{F}_q^n) \mid \mathrm{rk}(\phi_{\hat{\mathbf{v}}}) = R \quad .$$

The invariant function from section 5 then takes the form

$$(\phi, \hat{\mathbf{v}}) \overset{F_1}{\longmapsto} \phi/\hat{\mathbf{v}}.$$

Fix the choice of rank R and let k be the exponent such that $\|\mathbb{P}_{\phi,R}\| = q^k$. Assume that F restricted to S_R is distinguishing.

Let ϕ and ψ denote the two input trilinear forms with the existence of an $A \in \mathrm{GL}(n, \mathbb{F}_q)$ such that $\psi = \phi \circ A$ in question. Consider the relation set

$$R^{F_1}_{\phi,\psi} := \{(\mathbf{u}, \mathbf{v}) \in \mathbb{P}^2_{\phi,R} \mid F_1(\phi, \hat{\mathbf{u}}) = F_1(\psi, \hat{\mathbf{v}})\}.$$

If ϕ and ψ are not isomorphic, then neither are their restrictions to $\mathsf{ker}(\phi_{\hat{\mathbf{v}}}) \times \mathbb{F}_q^n \times \mathbb{F}_q^n$, implying $R^{F_1}_{\phi,\psi}$ is empty. If ϕ and ψ are isomorphic, by the distinguishing property of F_1, with high probability, $F_1(\phi, \hat{\mathbf{u}}) = F_1(\psi, \hat{\mathbf{v}})$ if and only if $\exists A \in \mathrm{GL}(n, q)$ such that $\psi = \phi \circ A$ and $A^{-1}\hat{\mathbf{u}} = \hat{\mathbf{v}}$.

The invariance and the distinguishing property of F_1 together ensure that with high probability, a random pair $(\hat{\mathbf{u}}, \hat{\mathbf{v}}) \in R^{F_1}_{\phi,\psi}$ is a witness to the isomorphism of ϕ and ψ restricted to $\mathsf{ker}(\phi_{\hat{\mathbf{u}}}) \times \mathbb{F}_q^n \times \mathbb{F}_q^n$. That is, there exists an $A \in \mathrm{GL}(n, q)$ such that $\hat{\mathbf{v}} = A^{-1}\hat{\mathbf{u}}$ and A moves the restriction of ϕ to the restriction of ψ. In particular, A restricted to $\mathsf{ker}(\phi_{\hat{\mathbf{u}}})$ acts in the first dimension. Therefore, with $(\hat{\mathbf{u}}, \hat{\mathbf{v}})$ as the partial information, the Gröbner basis algorithm of [20,42] becomes a heuristic polynomial time test of membership in $R^{F_1}_{\phi,\psi}$.

Invoke Szegedy's algorithm with X as $\mathbb{P}_{\phi,R}$, R as $R^{F_1}_{\phi,\psi}$, α as $1/3$ and uniformly sampling an $H \subset \mathbb{P}_{\phi,R}$ of size $\Theta\left(q^{k/3}\right)$. For isomorphic ϕ and ψ, there exists at least one $A_{\phi,\psi} \in \mathrm{GL}(n, q)$ such that $\psi = \phi \circ A_{\phi,\psi}$. Therefore, by the invariance and the distinguishing nature of F_1,

$$\Pr\left((R^{F_1}_{\phi,\psi} \cap (H \times H)) \neq \emptyset\right) \geq \Pr\left((H \cap A_{\phi,\psi}(H)) \neq \emptyset\right) \geq \Omega\left(q^{-k/3}\right),$$

proving that the probability gap may be taken to be $\epsilon = \Omega\left(q^{-k/3}\right)$. Therefore, for a rank parameter such that the sampling cost samp-cost is in polynomial time, the decision version of ATFE can be solved in $q^{k/3}\mathsf{poly}(n, \log q)$ time on a quantum computer. To tackle the promise search version ATFE within the same running time, applying the extensions of Szegedy's algorithm by Magniez, Nayak, Richter, Roland, and Santha [38,37], the search version ATFE can also be solved in

$$q^{k/3} \cdot \mathsf{poly}(n, \log q)$$

time on a quantum computer. Curiously, it is not obvious if the claw finding formalism in Tani's algorithm can be adapted to the low-rank birthday attacks. If we can efficiently derive canonical forms in addition to testing the isomorphism class of the restriction, then Tani's algorithm apply immediately. The reason being that we can order the canonical form representatives and obtain a comparison oracle.

6.4 Low-rank birthday attacks on MCE via quantum random walks

Recall the notation from section 4. We next phrase MCE as claw finding. Let ϕ, ψ be the two input isomorphic trilinear forms. Take X and Y as uniformly random subsets of co-rank 1 projective points, each of size $q^{n/2}$. Take f as the $\hat{\mathbf{u}} \longmapsto \bar{\phi}[\hat{\mathbf{u}}]$ map and g as the $\hat{\mathbf{u}} \longmapsto \bar{\psi}[\hat{\mathbf{u}}]$ map. The birthday paradox ensures that there is a solution to claw finding with constant positive probability. Invoking Tani's algorithm solves MCE in $q^{n/3}\mathsf{poly}(n, \log q)$ time.

Acknowledgements. We thank the anonymous reviewers for their careful reading and helpful suggestions. Youming Qiao was partly supported by ARC DP200100950 and LP220100332. Gang Tang was partly supported by ARC LP220100332, Sydney Quantum Academy, and EPSRC grant EP/V011324/1.

References

1. Alamati, N., Feo, L.D., Montgomery, H., Patranabis, S.: Cryptographic group actions and applications. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12492, pp. 411–439. Springer (2020). https://doi.org/10.1007/978-3-030-64834-3_14, `https://doi.org/10.1007/978-3-030-64834-3_14`
2. Ambainis, A.: Quantum walk algorithm for element distinctness. SIAM Journal on Computing **37**(1), 210–239 (2007). https://doi.org/10.1137/S0097539705447311, `https://doi.org/10.1137/S0097539705447311`
3. Babai, L.: Graph isomorphism in quasipolynomial time [extended abstract]. In: Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016. pp. 684–697 (2016)
4. Babai, L.: Canonical form for graphs in quasipolynomial time: preliminary report. In: Charikar, M., Cohen, E. (eds.) Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019. pp. 1237–1246. ACM (2019). https://doi.org/10.1145/3313276.3316356, `https://doi.org/10.1145/3313276.3316356`
5. Baldi, M., Barenghi, A., Beckwith, L., Biasse, J.F., Esser, A., Gaj, K., Mohajerani, K., Pelosi, G., Persichetti, E., Saarinen, M.J., Santini, P., Wallace, R.: Less: Linear equivalence signature scheme (2023), `https://www.less-project.com/LESS-2023-08-18.pdf`

6. Bardet, M., Bros, M., Cabarcas, D., Gaborit, P., Perlner, R.A., Smith-Tone, D., Tillich, J., Verbel, J.A.: Improvements of algebraic attacks for solving the rank decoding and minrank problems. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12491, pp. 507–536. Springer (2020). https://doi.org/10.1007/978-3-030-64837-4_17, https://doi.org/10.1007/978-3-030-64837-4_17
7. Bardet, M., Otmani, A., Saeed-Taha, M.: Permutation code equivalence is not harder than graph isomorphism when hulls are trivial. In: 2019 IEEE International Symposium on Information Theory (ISIT). pp. 2464–2468. IEEE (2019). https://doi.org/10.1109/ISIT.2019.8849855
8. Barenghi, A., Biasse, J.F., Ngo, T., Persichetti, E., Santini, P.: Advanced signature functionalities from the code equivalence problem. International Journal of Computer Mathematics: Computer Systems Theory **7**(2), 112–128 (2022)
9. Battagliola, M., Borin, G., Meneghetti, A., Persichetti, E.: Cutting the grass: Threshold group action signature schemes. Cryptology ePrint Archive (2023)
10. Belsley, E.: Rates of convergence of Markov chains related to association schemes, Harvard University Ph. D. Ph.D. thesis, thesis (1993)
11. Beullens, W.: Not enough less: An improved algorithm for solving code equivalence problems over f q. In: International Conference on Selected Areas in Cryptography. pp. 387–403. Springer (2020)
12. Beullens, W.: Graph-theoretic algorithms for the alternating trilinear form equivalence problem. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III. Lecture Notes in Computer Science, vol. 14083, pp. 101–126. Springer (2023). https://doi.org/10.1007/978-3-031-38548-3_4, https://doi.org/10.1007/978-3-031-38548-3_4
13. Biasse, J.F., Micheli, G., Persichetti, E., Santini, P.: Less is more: code-based signatures without syndromes. In: Progress in Cryptology-AFRICACRYPT 2020: 12th International Conference on Cryptology in Africa, Cairo, Egypt, July 20–22, 2020, Proceedings 12. pp. 45–65. Springer (2020)
14. Bläser, M., Chen, Z., Duong, D.H., Joux, A., Nguyen, N.T., Plantard, T., Qiao, Y., Susilo, W., Tang, G.: On digital signatures based on isomorphism problems: Qrom security, ring signatures, and applications. Cryptology ePrint Archive, Paper 2022/1184 (2022), https://eprint.iacr.org/2022/1184
15. 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
16. Bosma, W., Cannon, J., Playoust, C.: The Magma algebra system. I. The user language. J. Symbolic Comput. **24**(3-4), 235–265 (1997). https://doi.org/10.1006/jsco.1996.0125, http://dx.doi.org/10.1006/jsco.1996.0125, computational algebra and number theory (London, 1993)
17. Bouillaguet, C., Fouque, P., Véber, A.: Graph-theoretic algorithms for the "isomorphism of polynomials" problem. In: Advances in Cryptology - EUROCRYPT 2013. pp. 211–227 (2013)
18. Brassard, G., Hoyer, P., Tapp, A.: Quantum cryptanalysis of hash and claw-free functions. In: Lucchesi, C.L., Moura, A.V. (eds.) LATIN'98: Theoretical Informatics. pp. 163–169. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)

19. Brassard, G., Yung, M.: One-way group actions. In: Advances in Cryptology - CRYPTO 1990. pp. 94–107 (1990)
20. Bürgisser, P., Franks, C., Garg, A., de Oliveira, R.M., Walter, M., Wigderson, A.: Efficient algorithms for tensor scaling, quantum marginals, and moment polytopes. In: 59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018. pp. 883–897 (2018). https://doi.org/10.1109/FOCS.2018.00088
21. Chou, T., Niederhagen, R., Persichetti, E., Ran, L., Randrianarisoa, T.H., Reijnders, K., Samardjiska, S., Trimoska, M.: Matrix code equivalence digital signature (2023), `https://www.meds-pqc.org/spec/MEDS-2023-07-26.pdf`
22. 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: International Conference on Cryptology in Africa. pp. 28–52. Springer (2023)
23. Couvreur, A., Debris-Alazard, T., Gaborit, P.: On the hardness of code equivalence problems in rank metric. arXiv preprint arXiv:2011.04611 (2020)
24. D'Alconzo, G., Gangemi, A.: Trifors: Linkable trilinear forms ring signature. Cryptology ePrint Archive (2022)
25. Ducas, L., Postlethwaite, E.W., Pulles, L.N., Woerden, W.v.: Hawk: Module lip makes lattice signatures fast, compact and simple. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 65–94. Springer (2022)
26. Ducas, L., van Woerden, W.: On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 643–673. Springer (2022)
27. Faugère, J., Perret, L.: Polynomial equivalence problems: Algorithmic and theoretical aspects. In: Advances in Cryptology - EUROCRYPT 2006. pp. 30–47 (2006)
28. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Advances in Cryptology – CRYPTO 1986. pp. 186–194 (1986)
29. Fulman, J., Goldstein, L.: Stein's method and the rank distribution of random matrices over finite fields. The Annals of Probability **43**(3) (may 2015). https://doi.org/10.1214/13-aop889, `https://doi.org/10.1214%2F13-aop889`
30. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. J. ACM **38**(3), 691–729 (1991). https://doi.org/10.1145/116825.116852
31. Grochow, J.A., Qiao, Y.: On the complexity of isomorphism problems for tensors, groups, and polynomials I: tensor isomorphism-completeness. SIAM J. Comput. **52**(2), 568–617 (2023). https://doi.org/10.1137/21m1441110, `https://doi.org/10.1137/21m1441110`
32. Grochow, J.A., Qiao, Y., Tang, G.: Average-case algorithms for testing isomorphism of polynomials, algebras, and multilinear forms. journal of Groups, complexity, cryptology **14** (2022)
33. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219 (1996)
34. 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 - 17th International Conference, TCC 2019. vol. 11891, pp. 251–281. Springer (2019)

35. Kipnis, A., Shamir, A.: Cryptanalysis of the hfe public key cryptosystem by relinearization. In: Annual International Cryptology Conference. pp. 19–30. Springer (1999)
36. Leon, J.: Computing automorphism groups of error-correcting codes. IEEE Transactions on Information Theory **28**(3), 496–511 (1982)
37. Magniez, F., Nayak, A., Richter, P.C., Santha, M.: On the hitting times of quantum versus random walks. Algorithmica **63**, 91–116 (2012)
38. Magniez, F., Nayak, A., Roland, J., Santha, M.: Search via quantum walk. In: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing. pp. 575–584 (2007)
39. Reijnders, K., Samardjiska, S., Trimoska, M.: Hardness estimates of the code equivalence problem in the rank metric. Designs, Codes and Cryptography pp. 1–30 (2024)
40. Sendrier, N.: Finding the permutation between equivalent linear codes: The support splitting algorithm. IEEE Trans. Inf. Theory **46**(4), 1193–1203 (2000). https://doi.org/10.1109/18.850662, `https://doi.org/10.1109/18.850662`
41. Szegedy, M.: Spectra of quantized walks and a $\sqrt{\delta\epsilon}$-rule. arXiv preprint quant-ph/0401053 (2004)
42. Tang, G., Duong, D.H., Joux, A., Plantard, T., Qiao, Y., Susilo, W.: Practical post-quantum signature schemes from isomorphism problems of trilinear forms. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 582–612. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_21, `https://doi.org/10.1007/978-3-031-07082-2_21`
43. Tani, S.: Claw finding algorithms using quantum walk. Theoretical Computer Science **410**(50), 5285–5297 (2009)
44. Verbel, J., Baena, J., Cabarcas, D., Perlner, R., Smith-Tone, D.: On the complexity of "superdetermined" minrank instances. In: Post-Quantum Cryptography: 10th International Conference, PQCrypto 2019, Chongqing, China, May 8–10, 2019 Revised Selected Papers 10. pp. 167–186. Springer (2019)

A Low-rank point sampling via min-rank step

The sampling step can be done by either the min-rank method, or the graph-walking method. The graph-walking method involves q, so it works best for relatively small q. When q is large, the min-rank method is more effective. To use min-rank to do sampling requires a bit of twist, so we record the idea here.

Suppose we wish to sample a rank-r point $\hat{\mathbf{v}} \in \mathbb{P}(\mathbb{F}_q^n)$ for an alternating trilinear form ϕ, and suppose that there are q^k-many rank-r projective points for a random ϕ. To sample such points, we make a heuristic assumption that the first k coordinates of these rank-r points are in uniform random. Therefore, to sample one point, we can randomly choose the first k coordinates and then resort to the min-rank procedure.

More specifically, for $i \in [n]$, let A_i be the alternating matrix representing the bilinear form ϕ_{e_i}, where e_i is the ith standard basis vector. Let x_i, $i \in [n]$, be formal variables, and set $A = \sum_{i\in[n]} x_i A_i$. So for $i \in [1 \dots k]$, let $x_i = \alpha_i x_1$, where $\alpha_i \in_R \mathbb{F}_q$. This gives us a min-rank instance with $n-k$ matrices of size $n \times n$.

To estimate the min-rank cost, we use the algorithm from [6]. Consider an (n, K, r) minrank instance, namely finding a rank-r matrix in a linear span of K $n \times n$ matrices. First, we need to compute the smallest b such that $b < r+2$ and

$$\binom{n}{r}\binom{K+b-1}{b} - 1 \leq \sum_{i=1}^{b}(-1)^{i+1}\binom{n}{r+i}\binom{n+i-1}{i}\binom{K+b-i-1}{b-i}.$$

Based on this b, the complexity is estimated as

$$O\big(K \cdot (r+1) \cdot (\binom{n}{r} \cdot \binom{K+b-1}{b})^2\big).$$

For concrete values of n, $K = n-k$ and r, the above formulas allow for the estimation of the concrete security parameters.

Note that the min-rank instance above has some structural constraints due to alternating trilinear forms. As pointed out in [12], such structures should impact the min-rank algorithm from [6] adversely. Still, we use the estimates from [6] as they should serve as a lower bound. We also compare the estimates from [6] with the analysis of the Kipnis–Shamir modelling [35] in [44], and found the ones from [6] are lower.