Don't Use It Twice! Solving Relaxed Linear Equivalence Problems

Alessandro Budroni[1], Jesús-Javier Chi-Domínguez[1], Giuseppe D'Alconzo[2], Antonio J. Di Scala[2], and Mukul Kulkarni[1]

[1] Cryptography Research Center, Technology Innovation Institute, UAE
{alessandro.budroni,jesus.dominguez,mukul.kulkarni}@tii.ae
[2] Department of Mathematical Sciences, Polytechnic University of Turin, Italy
{giuseppe.dalconzo,antonio.discala}@polito.it

Abstract. The Linear Code Equivalence (LCE) Problem has received increased attention in recent years due to its applicability in constructing efficient digital signatures. Notably, the LESS signature scheme based on LCE is under consideration for the NIST post-quantum standardization process, along with the MEDS signature scheme that relies on an extension of LCE to the rank metric, namely the Matrix Code Equivalence (MCE) Problem. Building upon these developments, a family of signatures with additional properties, including linkable ring, group, and threshold signatures, has been proposed. These novel constructions introduce relaxed versions of LCE (and MCE), wherein multiple samples share the same secret equivalence. Despite their significance, these variations have often lacked a thorough security analysis, being assumed to be as challenging as their original counterparts. Addressing this gap, our work delves into the sample complexity of LCE and MCE — precisely, the sufficient number of samples required for efficient recovery of the shared secret equivalence. Our findings reveal, for instance, that one should not use the same secret twice in the LCE setting since this enables a polynomial time (and memory) algorithm to retrieve the secret. Consequently, our results unveil the insecurity of two advanced signatures based on variants of the LCE Problem.

Keywords: Algebraic Attack · Code Equivalence · Group Actions · Cryptanalysis · Post-quantum Cryptography

1 Introduction

Following the ongoing NIST post-quantum standardization process for additional digital signature schemes [28], there has been an increased interest in constructing new quantum-resistant digital signatures. Moving beyond the proposals at the prior NIST post-quantum standardization process [27], the research community explored a broader spectrum of computational problems, conjectured to be hard, for building efficient signature schemes. A family of such hard problems is represented by those computational problems consisting of finding an equivalence or isomorphism between two algebraic/geometrical structures. For example, among the candidates for the NIST post-quantum standardization process, the digital signature Hawk [12] relies on the hardness of the Lattice Isomorphism Problem (LIP), LESS [2] on the Linear Code Equivalence Problem (LCE), MEDS [15] on the Matrix Code Equivalence Problem (MCE), and SQIsign [14] on the problem of finding isogenies between supersingular elliptic curves.

All these hard problems can be modeled as group actions. This common framework has been utilized by cryptographers in two significant ways. First, protocols defined for a specific hard problem have often been adapted into analogous protocols using another hard problem, leveraging the similar structure and properties of the underlying group actions. For example, the Calamari ring signature [10] relying on isogenies has been translated to code equivalence [4]. Second, some protocols have been defined in a general manner for group actions and subsequently instantiated with specific problems [25,22,7]. This approach not only broadens the applicability of these cryptographic protocols but also provides a unified theoretical foundation for their security and efficiency.

While group actions used in cryptography are generally assumed to guarantee one-wayness, specific group actions might or might not satisfy certain additional properties such as weak-unpredictability and weak-pseudorandomness. This subject has already been studied for LCE and MCE by D'Alconzo and Di Scala [19] and for LIP by Benčina et al. [8]. Consequently, instantiating protocols with a specific group action without ensuring that stronger cryptographic properties are satisfied may result in insecure protocols. In addition, to achieve specific functionalities such as threshold signature or linkability on ring signatures, some relaxed versions of these hard problems have been proposed. These variants are often conjectured to exhibit a level of difficulty comparable to their original counterparts but without formal proof or comprehensive cryptanalytic investigation.

In this work, we significantly improve the sample complexity estimated by D'Alconzo and Di Scala for LCE and MCE, i.e., the sufficient number of samples sharing the same secret required for breaking weak-unpredictability. Furthermore, we give an algorithm to solve two variants of LCE, namely the Inverse Code Equivalence Problem (ILCE) and the Code Equivalence Problem with two samples (2-LCE),[3] in polynomial time, that were introduced to construct linkable ring signatures [4] and threshold signatures [7], respectively. As a consequence, the schemes that rely on the hardness of ILCE and 2-LCE are not secure. However, we wish to highlight that our result does not affect the one-wayness of LCE/MCE.

We summarize in Table 1 the applications of LCE and MCE group actions that are still considered secure, and the ones that have been discovered not secure by this work and [19] since they require stronger properties such as weak-

[3] The authors in [7] gave a more general problem definition in terms of group actions, namely 2-Group Action Inverse Problem (2-GAIP). Here, we refer by 2-LCE to the 2-GAIP from [7] instantiated with LCE.

unpredictability and weak-pseudorandomness. For the case of constructing linkable ring signatures using the inverse problem of MCE, our work reveals that, algebraically, this problem is significantly weaker than classic MCE. Thus, we believe that further investigation in this case is necessary.

	ID scheme / signature	Commitment	Linkable ring signature from [10,4,16]	Pseudo random function from [1]	Updatable encryption from [25]
LCE	✓	✓	✗	✗	✗
MCE	✓	✓	✓(?)	✗	✗

Table 1: Overview of the secure and insecure known instantiations of primitives constructed from LCE and MCE group actions. The symbols ✗ and ✓ denote that the corresponding primitive is insecure or remains secure. The symbol ✓(?) denotes that no specific attacks are known, but we suggest further investigation. The third column in the LCE setting concerns the cryptographic scenario when the code length doubles the code dimension.

1.1 Overview of the contribution

Informally, we say that two linear codes $\mathcal{C}_1$ and $\mathcal{C}_2$ of length n and dimension k over a finite field $\mathbb{F}_q$ are equivalent if there exists a monomial matrix $\boldsymbol{Q} \in \mathbb{F}_q^{n\times n}$ such that $\mathcal{C}_2 = \mathcal{C}_1\boldsymbol{Q}$. Given two generators $\boldsymbol{G}_1, \boldsymbol{G}_2 \in \mathbb{F}_q^{k\times n}$ of two equivalent codes, the Linear Code Equivalence Problem (LCE) is the problem of finding an invertible matrix $\boldsymbol{S}$ and a monomial matrix $\boldsymbol{Q}$ such that $\boldsymbol{G}_2 = \boldsymbol{S}\boldsymbol{G}_1\boldsymbol{Q}$. When $\boldsymbol{Q}$ is a permutation matrix, the problem is called Permutation Code Equivalence Problem (PCE).

On the hardness of LCE given t samples

Our first contribution is investigating the impact of providing more than one LCE sample sharing the same secret matrix $\boldsymbol{Q}$ to the adversary. We explore how this explicitly affects the hardness of recovering $\boldsymbol{Q}$. In particular, we derive a concrete bound on the number of necessary samples that allow an efficient recovery of the secret. Additionally, similar results are also obtained for MCE. We present our result in the following lemma:

Lemma (Informal). *For (n, k)-linear codes over a finite field $\mathbb{F}_q$, the secret monomial matrix $\boldsymbol{Q}$ can be recovered from $\left\lfloor \frac{n^2}{k(n-k)} \right\rfloor + 1$ samples of* LCE *sharing the same $\boldsymbol{Q}$ in polynomial time, with non-negligible probability.*

The above result improves upon the work by D'Alconzo and Di Scala [19], who provided a bound of $n \cdot k$ samples applicable solely to code generators that are not in systematic form. In contrast, our result removes this limitation, extending the applicability to codes represented in systematic form as well. The key ingredient of our result relies on constructing a linear system from each sample, where only the entries of $\boldsymbol{Q}$ are the variables (and not those of $\boldsymbol{S}$). Specifically, we use the relation $\boldsymbol{G}_1\boldsymbol{Q}\boldsymbol{H}_2^\top = 0$, where $\boldsymbol{H}_2$ is a parity-check matrix of $\boldsymbol{G}_2$, to construct the following homogeneous linear system:

$$(\boldsymbol{G}_1 \otimes \boldsymbol{H}_2) \cdot \mathsf{vec}(\boldsymbol{Q}) = \boldsymbol{0}, \tag{1}$$

where $\mathsf{vec}(\boldsymbol{Q})$ is the column vector whose entries are the entries of $\boldsymbol{Q}$ row-by-row. This linear system is underdetermined, meaning that there are fewer equations than variables. However, by combining the systems from different samples, we obtain with high probability a determined linear system whose solution can be found via Gaussian elimination, leading to the recovery of $\boldsymbol{Q}$.

Solving 2-LCE and ILCE for $(2k, k)$-linear codes

Our second contribution is to introduce a polynomial-time algorithm for solving 2-LCE, i.e., the problem of retrieving $\boldsymbol{Q}$ from only 2 LCE samples, specifically for $k = n/2$. Thanks to the fact that ILCE can be seen as a 2-LCE instance via a simple transformation, we are able to solve this problem in polynomial time as well. Our algorithm is inspired by Saeed's work [33] and, in addition to the results mentioned in the lemma above, it exploits the structure of the secret monomial matrix to recover it.

Our method consists of first constructing a linear system as in Equation (1) with the two available LCE samples

$$\begin{bmatrix} \boldsymbol{G}_1 \otimes \boldsymbol{H}_2 \\ \boldsymbol{G}_1' \otimes \boldsymbol{H'}_2 \end{bmatrix} \cdot \mathsf{vec}(\boldsymbol{Q}) = \boldsymbol{0}. \tag{2}$$

Then, we guess which entries of the secret matrix $\boldsymbol{Q}$ are non-zero. Thanks to the structure of $\boldsymbol{Q}$, for each guess on a non-zero variable, we can simultaneously guess additional $2n-2$ entries on the same row and column to be zero. When evaluating the variables corresponding to our guess in the system in Equation (2), we obtain a new smaller non-homogeneous system of the form $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$. We prove that the matrix of coefficients $\boldsymbol{A}$ is not full-rank, allowing us to distinguish correct guesses from the wrong ones, with high probability, using the Rouché–Capelli Theorem: to determine whether the obtained systems accept solutions or not, we check whether $\operatorname{rank}(\boldsymbol{A}) = \operatorname{rank}(\boldsymbol{A}|\boldsymbol{b})$. We show that for wrong guesses $\operatorname{rank}(\boldsymbol{A}) \neq \operatorname{rank}(\boldsymbol{A}|\boldsymbol{b})$ with probability $1 - \frac{1}{q}$, which allows us to distinguish them efficiently from correct guesses.

Our algorithm consists of iterating this test on all possible n^2 guesses on the entries of $\boldsymbol{Q}$ and setting the variables that did not pass the Rouché–Capelli test to zero. Our heuristic analysis shows that, for $q > 2$ and $n \geq 4$, we are able to discard enough variables so that the remaining ones can be retrieved via

Gaussian elimination, hence revealing the secret $\boldsymbol{Q}$. The time complexity of our algorithm is indeed polynomial and consists of making two rank computations for each of the n^2 guesses, resulting in $O(n^{2+2\omega})$, for $\omega \in [2,3]$.

We validate our theoretical results through extensive experiments and simulations performed by means of a SageMath [38] proof-of-concept implementation. All scripts are available in [13].

1.2 Related work

Permutation Code Equivalence. The cryptanalysis of equivalence problems on linear codes started with Leon's algorithm [24], which presented a way to compute the permutation between two equivalent codes using the information provided by codewords of minimal weight, but it is unpractical for cryptographic instances. Later, Petrank and Roth [31] showed that PCE is unlikely to be NP-complete. In his seminal work [36], Sendrier introduced the Support Splitting Algorithm, which can recover the secret permutation underlying PCE in time $\widetilde{O}(q^h)$, where q is the cardinality of the field and h is the dimension of the *hull* of the code, namely the intersection between the code and its dual. In addition, two more attacks on PCE with trivial hulls have been proposed [33,3]. All these results imply that PCE is not hard when the hull is small, and this happens with high probability when the code is randomly chosen ([35] showed that in this case, the hull dimension is a small constant). Hence, PCE must be instantiated with self-dual or weakly-dual codes to be suitable in cryptography.

Linear Code Equivalence and Matrix Code Equivalence. In [37] Sendrier and Simos showed that LCE can be reduced to PCE using the closure of the code. This implied that one should be able to solve LCE using the above techniques, but, for $q \geq 5$, the closure of a code is always weakly-self dual, and the Support Splitting Algorithm becomes unfeasible. Contrary to PCE, random instances of LCE remain intractable, and hence, they can be used in the design of cryptosystems. After the publication of LESS [11], the effort for cryptanalyzing PCE and LCE increased [5,9], which led to a refinement of the conjectured practical complexity of solving these problems. Recently, [17] showed that the LCE (as well as PCE) is equivalent to its variant which is based on canonical form of the underlying codes. In some parameter regimes, they provided the best known attacks on LCE. In summary, the known techniques are practical for particular classes of codes, while finding the permutation or the linear map leading to the equivalence seems to be still intractable for carefully generated instances. In the case of matrix codes, the equivalence problem was first studied from a cryptographic point of view in [32] and it is further cryptanalyzed in the work that introduces MEDS [16], presenting an adaptation of Leon's algorithm in the setting of matrix codes and an algebraic modelling.

Organization. We give in Section 2 the necessary notation and preliminaries. In Section 3 we give results on the sample complexity of LCE and MCE. In Section 4 we describe a new algorithm that solves both ILCE and 2-LCE in polynomial time, and we give the result of our experiments related to it in Section 5. Finally, we discuss the cryptographic implications of our work in Section 6.

2 Preliminaries

2.1 Notation

In this paper, we denote with $\mathbb{N}$, $\mathbb{Z}$ and $\mathbb{R}$ the sets of natural, integer and real numbers respectively. For a number $n \in \mathbb{N}$ we use $[n]$ for the set $\{1, 2, \ldots, n\}$. We denote matrices with upper-case bold letters (e.g. $\boldsymbol{A}$) and vectors with lower-case bold letters (e.g. $\boldsymbol{a}$). We treat vectors as columns unless otherwise specified. Let $\mathbb{F}_q$ denote a finite field of order q. The tensor product $(\boldsymbol{A} \otimes \boldsymbol{B}) \in \mathbb{F}_q^{mr \times ns}$ of two matrices $\boldsymbol{A} \in \mathbb{F}_q^{m \times n}$ and $\boldsymbol{B} \in \mathbb{F}_q^{r \times s}$ is defined as the Kronecker product of $\boldsymbol{A}$ and $\boldsymbol{B}$.

We use $\mathsf{GL}_n(\mathbb{F}_q)$ for the set of invertible $n \times n$ matrices with elements in $\mathbb{F}_q$, $\mathsf{Perm}_n(\mathbb{F}_q)$ for the set of permutation matrices of dimension n, and $\mathsf{Mono}_n(\mathbb{F}_q)$ for the set of $n \times n$ *monomial* matrices, i.e., that can be written as $\boldsymbol{M} = \boldsymbol{D}\boldsymbol{P}$, where $\boldsymbol{D} \in \mathbb{F}_q^{n \times n}$ is full-rank diagonal, and $\boldsymbol{P} \in \mathsf{Perm}_n(\mathbb{F}_q)$. We also use $\boldsymbol{I}_n$ to denote $n \times n$ identity matrix over $\mathbb{F}_q$.

For any matrix $\boldsymbol{M} \in \mathbb{F}_q^{m \times n}$, we write $\mathsf{vec}(\boldsymbol{M})$ to denote the column vector of mn coefficients consisting of the concatenation of the rows of $\boldsymbol{M}$.

We assume that computing multiplication and inverse of matrices can be performed using $O(n^\omega)$ field operations for some $\omega \in [2, 3]$.[4] Consequently, we assume that solving a linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ with $\boldsymbol{A} \in \mathbb{F}_q^{n \times n}$ and $\boldsymbol{b} \in \mathbb{F}_q^n$ takes time $O(n^\omega)$ field operations, and that calculating the rank (and kernel) of $\boldsymbol{A} \in \mathbb{F}_q^{n \times n}$ costs $O(n^\omega)$ field operations.[5]

The following propositions will be used in Section 4.

Proposition 1. *Let $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \in \mathbb{F}_q^{(k) \times (2k-1)}$ be matrices, for $k \geq 2$. Then the rank of the matrix, $\boldsymbol{M} \in \mathbb{F}_q^{(2k^2) \times (2k-1)^2}$, defined as*

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{A} \otimes \boldsymbol{B} \\ \boldsymbol{C} \otimes \boldsymbol{D} \end{bmatrix}$$

is strictly smaller than $2k^2$.

Proof. Given the dimension of the matrices, there exist $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta} \in \mathbb{F}_q^k$ non-zero vectors such that

$$\begin{bmatrix} \boldsymbol{\alpha} \; \boldsymbol{\gamma} \end{bmatrix} \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{C} \end{bmatrix} = 0, \qquad \text{and} \qquad \begin{bmatrix} \boldsymbol{\beta} \; \boldsymbol{\delta} \end{bmatrix} \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{D} \end{bmatrix} = 0.$$

Then, the vector

$$\boldsymbol{v} = (\boldsymbol{\alpha} \otimes \boldsymbol{\beta}, -\boldsymbol{\gamma} \otimes \boldsymbol{\delta})$$

[4] For example, in the case of the well-known Strassen's algorithm which is considered as the best algorithm for matrix multiplications for large n, one can set $\omega = \log_2(7)$.

[5] If the matrix $\boldsymbol{A} \in \mathbb{F}_q^{r \times s}$ is rectangular, we set $n = \max\{r, s\}$ in the complexity.

is such that $\boldsymbol{v} \cdot \boldsymbol{M} = 0$. Indeed

$$(\boldsymbol{\alpha} \otimes \boldsymbol{\beta}, -\boldsymbol{\gamma} \otimes \boldsymbol{\delta}) \begin{bmatrix} \boldsymbol{A} \otimes \boldsymbol{B} \\ \boldsymbol{C} \otimes \boldsymbol{D} \end{bmatrix} = (\boldsymbol{\alpha} \cdot \boldsymbol{A}) \otimes (\boldsymbol{\beta} \cdot \boldsymbol{B}) - (\boldsymbol{\gamma} \cdot \boldsymbol{C}) \otimes (\boldsymbol{\delta} \cdot \boldsymbol{D}) =$$
$$(\boldsymbol{\gamma} \cdot \boldsymbol{C}) \otimes (\boldsymbol{\delta} \cdot \boldsymbol{D}) - (\boldsymbol{\gamma} \cdot \boldsymbol{C}) \otimes (\boldsymbol{\delta} \cdot \boldsymbol{D}) = 0$$

Hence, the left kernel of $\boldsymbol{M} \in \mathbb{F}_q^{2k^2 \times (2k-1)^2}$ is not null and it follows that $\operatorname{rank}(\boldsymbol{M}) < 2k^2$. □

2.2 Linear Codes and Equivalence Problems

An (n, k)-linear code $\mathcal{C}$ over $\mathbb{F}_q$ is a k-dimensional vector subspace of $\mathbb{F}_q^n$. We say that $\mathcal{C}$ has length n and dimension k. The *rate* of the code is the ratio $r := \frac{k}{n}$. Unless differently specified, along this paper we consider $r \in (0, \frac{1}{2}]$.

A matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$ is called a *generator matrix* of $\mathcal{C}$ if its rows form a basis of $\mathcal{C}$, that is $\mathcal{C} = \{\boldsymbol{u}^T \boldsymbol{G}, \boldsymbol{u} \in \mathbb{F}_q^k\}$. We say that $\boldsymbol{G}$ is in *systematic form* if $\boldsymbol{G} = (\boldsymbol{I}_k | \boldsymbol{M})$ for some $\boldsymbol{M} \in \mathbb{F}_q^{k \times (n-k)}$. A code that admits a generator in systematic form is called systematic code, and such generator can be obtained in polynomial-time by computing the reduced row-echelon form of a given generator. We denote this operation with $\mathsf{SF}(\cdot)$. Moreover, the generator in systematic form gives a standard basis for the k-dimensional vector subspace of $\mathbb{F}_q^n$ corresponding to the code. For a generator matrix $\boldsymbol{G}$, we denote $(\boldsymbol{G})_{-i}$ the generator matrix of the code punctured at position i, i.e., the code obtained by removing the i-th column from $\boldsymbol{G}$.

A full-rank matrix $\boldsymbol{H} \in \mathbb{F}_q^{(n-k) \times n}$ is called *parity check matrix* of $\mathcal{C}$ if and only if $\forall \boldsymbol{c} \in \mathcal{C}$ it holds that $\boldsymbol{H}\boldsymbol{c} = \boldsymbol{0}$. Note that if $\mathsf{SF}(\boldsymbol{G}) = (\boldsymbol{I}_k | \boldsymbol{M})$, for a matrix $\boldsymbol{M} \in \mathbb{F}_q^{k \times (n-k)}$, then the matrix $(-\boldsymbol{M}^\top | \boldsymbol{I}_{n-k})$ is a parity-check for $\mathcal{C}$. The parity-check matrix generates the *dual code* of $\mathcal{C}$, denoted with $\mathcal{C}^\perp$. The *hull* of a code $\mathcal{C}$ is defined as the intersection of $\mathcal{C}$ with its dual. A code $\mathcal{C}$ is said *weakly self-dual* if $\mathcal{C} \subset \mathcal{C}^\perp$ and *self-dual* if $\mathcal{C} = \mathcal{C}^\perp$. In both these cases, the dimension of the hull is equal to the dimension of the code.

Due to the extended variety of namings to the Linear Code Equivalence Problem (see Table 2), and for consistency between notations in different articles, we use the acronyms from [37] and [16].

Let $\boldsymbol{G}, \boldsymbol{G}'$ be the generator matrices of two (n, k)-linear codes $\mathcal{C}, \mathcal{C}'$. We say that $\mathcal{C}$ and $\mathcal{C}'$ are *equivalent* if there exist $\boldsymbol{S} \in \mathsf{GL}_k(\mathbb{F}_q)$ and $\boldsymbol{Q} \in \mathsf{Mono}_n(\mathbb{F}_q)$ such that $\boldsymbol{G}' = \boldsymbol{S}\boldsymbol{G}\boldsymbol{Q}$.

Definition 1 (Linear Code Equivalence (LCE) Problem). *Let $\boldsymbol{G}, \boldsymbol{G}' \in \mathbb{F}_q^{k \times n}$ be the generator matrices of two (n, k)-linear codes $\mathcal{C}, \mathcal{C}'$, respectively. The Code Equivalence Problem is to find matrices $\boldsymbol{S} \in \mathsf{GL}_k(\mathbb{F}_q)$ and $\boldsymbol{Q} \in \mathsf{Mono}_n(\mathbb{F}_q)$ (if they exist) such that $\boldsymbol{G}' = \boldsymbol{S}\boldsymbol{G}\boldsymbol{Q}$.*

Sometimes, in the literature, LCE is stated as in Definition 1 but with the assurance that such matrices $\boldsymbol{S}$ and $\boldsymbol{Q}$ establishing the equivalence between

	Permutation Code Equivalence Problem	Linear Code Equivalence Problem	Matrix Code Equivalence Problem
[37,23,29]	PCE	LCE	—
[34]	PEP	—	—
[17,30,2]	PEP	LEP	—
[7]	PEP	LEP	MCE
[16,32]	—	—	MCE

Table 2: Notation naming for the Linear, Permutation, and Matrix Code Equivalence Problems through the state-of-the-art.

the two codes exist. Indeed, cryptographic schemes inherently guarantee the equivalence by construction. Consequently, this work explicitly addresses and incorporates this scenario.

If instead of being a monomial, the secret matrix $\boldsymbol{Q}$ is a permutation matrix, then the problem is known as **Permutation Code Equivalence** (PCE) **Problem**.

A $(m \times r, k)$ *matrix code* is a subspace $\mathcal{D}$ of dimension k of the space of $m \times r$ matrices. The following problem was introduced in [32,16]. Two matrix codes $\mathcal{D}, \mathcal{D}'$ are *equivalent* if there exists two matrices $\boldsymbol{A} \in \mathsf{GL}_m(\mathbb{F}_q)$ and $\boldsymbol{B} \in \mathsf{GL}_r(\mathbb{F}_q)$ such that $\mathcal{D}' = \boldsymbol{A}\mathcal{D}\boldsymbol{B}$. In fact, [16, Lemma 1] proved that the MCE problem can be redefined in terms of the tensor product $\boldsymbol{A}^\top \otimes \boldsymbol{B}$ as described below.

Definition 2 (Matrix Code Equivalence (MCE) Problem). *Let $\boldsymbol{G}, \boldsymbol{G}' \in \mathbb{F}_q^{k \times mr}$ be generators of two $(m \times r, k)$-matrix codes $\mathcal{D}, \mathcal{D}'$ respectively. The Matrix Code Equivalence problem is to find (if they exist) $\boldsymbol{S} \in \mathsf{GL}_k(\mathbb{F}_q)$, $\boldsymbol{A} \in \mathsf{GL}_m(\mathbb{F}_q)$ and $\boldsymbol{B} \in \mathsf{GL}_r(\mathbb{F}_q)$ such that $\boldsymbol{G}' = \boldsymbol{S}\boldsymbol{G}(\boldsymbol{A}^\top \otimes \boldsymbol{B})$.*

Inverse Linear Code Equivalence Problem: In the context of linkable ring signatures, the following problem was initially introduced in [4].

Definition 3 (Inverse Linear Code Equivalence (ILCE) Problem). *Let $\boldsymbol{G}, \boldsymbol{G}', \boldsymbol{G}'' \in \mathbb{F}_q^{k \times n}$ be the generator matrices of three (n, k)-linear codes $\mathcal{C}$, $\mathcal{C}'$ and $\mathcal{C}''$ respectively. The Inverse Linear Code Equivalence Problem is to find (if they exist) $\boldsymbol{S} \in \mathsf{GL}_k(\mathbb{F}_q)$ and $\boldsymbol{Q} \in \mathsf{Mono}_n(\mathbb{F}_q)$ such that $\boldsymbol{G}' = \boldsymbol{S}\boldsymbol{G}\boldsymbol{Q}$ and $\boldsymbol{G}'' = \boldsymbol{S}^{-1}\boldsymbol{G}\boldsymbol{Q}^{-1}$.*

Similarly, we define the **Inverse Permutation Code Equivalence** (IPCE) **Problem** variant for when the secret monomial is a permutation matrix. There is also an Inverse Matrix Code Equivalence Problem variant, named IMCE and introduced in [16], that essentially replaces $\boldsymbol{Q} \in \mathsf{Mono}_r(\mathbb{F}_q)$ with $\boldsymbol{Q} \in \mathsf{GL}_{mr}(\mathbb{F}_q)$.

Definition 4 (Inverse Matrix Code Equivalence (IMCE) Problem). *Let $\boldsymbol{G}, \boldsymbol{G}', \boldsymbol{G}'' \in \mathbb{F}_q^{k \times mr}$ be generators of three $(m \times r, k)$-matrix codes $\mathcal{D}, \mathcal{D}'$ and $\mathcal{D}''$*

respectively. The Inverse Matrix Code Equivalence problem is to find (if they exist) $\boldsymbol{S} \in \mathsf{GL}_k(\mathbb{F}_q)$, $\boldsymbol{A} \in \mathsf{GL}_m(\mathbb{F}_q)$ and $\boldsymbol{B} \in \mathsf{GL}_r(\mathbb{F}_q)$ such that $\boldsymbol{G}' = \boldsymbol{SGQ}$ and $\boldsymbol{G}'' = \boldsymbol{S}^{-1}\boldsymbol{GQ}^{-1}$ with $\boldsymbol{Q} = (\boldsymbol{A}^\top \otimes \boldsymbol{B}) \in \mathsf{GL}_{mr}(\mathbb{F}_q)$.

Remark 1. In practice, one often works with generator matrices in systematic forms [2,15]. Hence, when $\boldsymbol{G}, \boldsymbol{G'}$ are in systematic form, we say that $\mathcal{C}, \mathcal{C}'$ are equivalent if there exists $\boldsymbol{Q} \in \mathsf{Mono}_n(\mathbb{F}_q)$ such that $\boldsymbol{G'} = \mathsf{SF}(\boldsymbol{GQ})$. The problems LCE, PCE, MCE, and the corresponding inverse variants can all be equivalently restated with the generators in systematic form without changing the hardness of the problems. Unless differently stated, we consider these problems in their systematic form version to ease the analysis presented in this paper.

2.3 Code equivalence problems with multiple samples

In order to study the stronger cryptographic properties of the equivalence problems, we introduce some new definitions allowing an interaction with stronger adversaries. We give in Definition 5 a relaxed version of LCE where the adversary has access to multiple LCE samples for the same secret monomial $\boldsymbol{Q}$.

Definition 5 (t-LCE). *Let n, k, q be positive integers such that $k < n$ and q is prime. Let $\boldsymbol{Q} \in \mathsf{Mono}_n(\mathbb{F}_q)$ be a* secret *monomial matrix. We denote by $\mathcal{L}_{n,k,q,\boldsymbol{Q}}$ the probability distribution on $\mathbb{F}_q^{k\times n} \times \mathbb{F}_q^{k\times n}$ obtained by sampling $\boldsymbol{M} \in \mathbb{F}_q^{k\times(n-k)}$ uniformly at random, setting $\boldsymbol{G} = (\, \boldsymbol{I}_k | \boldsymbol{M} \,) \in \mathbb{F}_q^{k\times n}$, and returning*

$$(\boldsymbol{G}, \boldsymbol{G}' = \mathsf{SF}(\boldsymbol{GQ})).$$

Given t independent samples from $\mathcal{L}_{n,k,q,\boldsymbol{Q}}$, the t-samples LCE problem*, denoted as t-LCE, is to find $\boldsymbol{Q}$.*

Informally, the distribution $\mathcal{L}_{n,k,q,\boldsymbol{Q}}$ samples a generator matrix $\boldsymbol{G}$ (in systematic form) of a random (n, k)-linear code over $\mathbb{F}_q$ and outputs the pair $(\boldsymbol{G}, \boldsymbol{G}')$, where $\boldsymbol{G}'$ is the generator matrix (in systematic form) of another equivalent linear code, and the equivalence is established via a secret monomial matrix $\boldsymbol{Q}$. When the parameters n, k, q are clear by the context, we simplify the notation and drop the indices from the shortening of the problem, i.e., we simply write t-LCE. Also, notice that 1-LCE corresponds to LCE, so in this case only write LCE. The t-*samples* version problem for ILCE is as follows.

Definition 6 (t-ILCE). *Let n, k, q be positive integers such that $k < n$ and q is prime. Let $\boldsymbol{Q} \in \mathsf{Mono}_n(\mathbb{F}_q)$ be a* secret *monomial matrix. We denote by $\widehat{\mathcal{L}}_{n,k,q,\boldsymbol{Q}}$ the probability distribution on $\mathbb{F}_q^{k\times n} \times \mathbb{F}_q^{k\times n} \times \mathbb{F}_q^{k\times n}$ obtained by sampling $\boldsymbol{M} \in \mathbb{F}_q^{k\times(n-k)}$ uniformly at random, setting $\boldsymbol{G} = (\, \boldsymbol{I}_k | \boldsymbol{M} \,) \in \mathbb{F}_q^{k\times n}$, and returning*

$$(\boldsymbol{G}, \boldsymbol{G}' = \mathsf{SF}(\boldsymbol{GQ}), \boldsymbol{G}'' = \mathsf{SF}(\boldsymbol{GQ}^{-1})).$$

Given t independent samples from $\widehat{\mathcal{L}}_{n,k,q,\boldsymbol{Q}}$, the t-samples ILCE problem*, denoted as t-ILCE, is to find $\boldsymbol{Q}$.*

Similarly, we call t-PCE (resp. t-IPCE) the problem of retrieving the secret matrix $\boldsymbol{P} \in \mathsf{Perm}_n(\mathbb{F}_q)$ given t samples of PCE (resp. IPCE). We also refer to t-MCE (resp. t-IMCE) the problem of retrieving the secret matrices $\boldsymbol{A} \in \mathsf{GL}_m(\mathbb{F}_q)$ and $\boldsymbol{B} \in \mathsf{GL}_r(\mathbb{F}_q)$, from t samples of MCE (resp. MCE).

2.4 Code equivalences modeled as group actions

A group action is a mapping of the form $\star : G \times X \to X$, where G is a group and X is a set, such that for any $g_1, g_2 \in G$ and any $x \in X$, we have $g_1 \star (g_2 \star x) = (g_1 g_2) \star x$. Cryptographic group actions are endowed with certain hardness properties, such as *one-wayness*, *weak-unpredictability* and *weak-pseudorandomness* [1].

The Linear Code Equivalence problem (Definition 1) has been modeled as a group action in [19] with the base set being $\mathbb{F}_q^{k \times n}$ and the group being $\mathsf{GL}_k(\mathbb{F}_q) \times \mathsf{Mono}_n(\mathbb{F}_q)$. In this work, we follow the approach similar to [7] that makes use of the systematic form. Recall that with $\mathsf{SF}()$ we denote the computation of reduced row-echelon form. Define the following equivalence relation

$$A \simeq_{\mathsf{SF}} B \iff \mathsf{SF}(A) = \mathsf{SF}(B), \qquad A, B \in \mathbb{F}_q^{k \times n}.$$

Consider the base set as $X = \mathbb{F}_q^{k \times n} / \simeq_{\mathsf{SF}}$ and the group as $G = \mathsf{Mono}_n(\mathbb{F}_q)$. Then the group action $\star$ is defined as

$$\star \colon G \times X \to X, \quad (\boldsymbol{Q}, \boldsymbol{G}) \mapsto \boldsymbol{Q} \star \boldsymbol{G} := \mathsf{SF}(\boldsymbol{G}\boldsymbol{Q}).$$

Similarly, PCE and MCE are modeled as group actions following the same framework. Consequently, it follows that LCE, PCE, and MCE are instances of the so-called Vectorization Problem [18].

2-LCE, 2-PCE, and 2-MCE are special cases of the 2-GAIP defined in [7, Problem 3]. Additionally, Definition 7 describes a useful property required for building secure threshold signatures as analyzed in [7].

Definition 7. (2-weakly pseudorandom group action [7, Def. 3]) *A group action $\star \colon G \times X \to X$ is 2-weakly pseudorandom if there is no probabilistic polynomial time algorithm that given $(x, g \star x)$ can distinguish with non-negligible probability between (x', y') and $(x', g \star x')$ with. $x', y' \in X$ sampled uniformly at random from X.*

3 Solving Code Equivalence with Multiple Instances

Recently, D'Alconzo and Di Scala [19] showed that, using representation theory, for certain group actions $(G, X, \star)$ it is possible to recover the secret $g \in G$ from a polynomial number of samples of the form $(x_i, g \star x_i)$ for random $x_i \in X$. In the case of the group action defined in Section 2.4, this can be viewed as variants of the problems t-LCE, t-PCE, and t-MCE that do not use the systematic form SF. They show that these variants can be solved efficiently (with high probability) when $t \in \mathsf{poly}(\lambda)$. In the case of t-LCE they showed that $t \geq nk$ samples are sufficient to recover the secret matrices $\boldsymbol{S}$ and $\boldsymbol{Q}$ (with high probability).

In this section, we improve the state-of-the-art by (a) showing that a significantly lower number of samples is sufficient to recover the secret matrix for the corresponding computational problem, and (b) unlike [19] our results cover even the cases when the codes are represented in the systematic form. In the rest of the paper we focus on the representation with codes in the systematic form since it leads to simpler analysis, however, we emphasize that our results extend to the general case as well since we can always compute the reduced row echelon form of the generator matrices.

In what follows, we focus our analysis on t-LCE. The main difference between t-LCE and t-MCE problems in the context of our techniques is that the secret matrix $\boldsymbol{Q}$ is a monomial matrix in the case of t-LCE, whereas for t-MCE problem the secret matrix is a tensor product $\left(\boldsymbol{A}^\top \otimes \boldsymbol{B}\right)$. Since we do not exploit the monomial structure of $\boldsymbol{Q}$ in the following analysis of t-LCE presented in this section, our results extend in a straightforward manner to the more general case of t-MCE. Moreover, we also do not restrict the underlying linear codes to possess any specific properties or structure e.g. self-dual codes or low dimension of hull. Our strategy, analogous to [19], consists of using the available samples to construct a linear system whose unknowns are the entries of the secret matrix. If the rank of the resulting linear system is large enough, then one is able to retrieve the secret simply via Gaussian elimination in polynomial time.

Proposition 2. *Given two generator matrices* $\boldsymbol{G} = (\,\boldsymbol{I}_k \big| \boldsymbol{M}\,) \in \mathbb{F}_q^{k\times n}$ *and* $\boldsymbol{G}' = \mathsf{SF}(\boldsymbol{G}\boldsymbol{Q}) = (\,\boldsymbol{I}_k \big| \boldsymbol{M}'\,) \in \mathbb{F}_q^{k\times n}$ *of two equivalent codes for some* $\boldsymbol{Q} \in \mathsf{GL}_n(\mathbb{F}_q)$*, we have that*

$$\Big[(\,\boldsymbol{I}_k \big| \boldsymbol{M}\,) \otimes (\,-\boldsymbol{M}'^\top \big| \boldsymbol{I}_{n-k}\,)\Big]\, \mathsf{vec}(\boldsymbol{Q}) = \boldsymbol{0}. \tag{3}$$

Proof. This is a straightforward application of [33, Definition 1.1.3, Corollary 3.2.13, and Corollary 3.2.20] without assuming the matrix $\boldsymbol{Q}$ to be a permutation, where $\boldsymbol{G} = (\,\boldsymbol{I}_k \big| \boldsymbol{M}\,)$ and the parity-check matrix of the code generated by $\boldsymbol{G}' = (\,\boldsymbol{I}_k \big| \boldsymbol{M}'\,)$ is given by $(\,-\boldsymbol{M}'^\top \big| \boldsymbol{I}_{n-k}\,)$. Note that for any code generated by matrix $\boldsymbol{G}'$, by definition we know that $\boldsymbol{G}'\boldsymbol{H}'^\top = \boldsymbol{0}$. We can therefore, obtain eq. (3) by substituting $\boldsymbol{G}\boldsymbol{Q}$ for $\boldsymbol{G}'$ and writing the resulting equation in tensor product notation. □

Notice that Proposition 2 gives $k(n-k)$ linear equations in the n^2 variables $\mathsf{vec}(\boldsymbol{Q})$ determining the entries of $\boldsymbol{Q}$. [6] Such a linear system has the following particular structure. Let us denote the (i,j)-th entry of $\boldsymbol{M}$ by $\boldsymbol{M}_{i,j}$, then the homogeneous linear system of equations derived from Equation (3) can be written as: $\boldsymbol{A} \cdot \mathsf{vec}(\boldsymbol{Q}) = \boldsymbol{0}$ where $\boldsymbol{A}$ is equal to

[6] In case of LCE we restrict $\boldsymbol{Q}$ to be in $\mathsf{Mono}_n(\mathbb{F}_q)$, while for MCE we assume that $n = mr$ and $\boldsymbol{Q} = \boldsymbol{A}^\top \otimes \boldsymbol{B}$ for some $\boldsymbol{A} \in \mathsf{GL}_m(\mathbb{F}_q)$ and $\boldsymbol{B} \in \mathsf{GL}_r(\mathbb{F}_q)$.

$$\begin{bmatrix} -\boldsymbol{M'}^\top & \boldsymbol{I}_c & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & -\boldsymbol{M}_{1,1}\boldsymbol{M'}^\top & \boldsymbol{M}_{1,1}\boldsymbol{I}_c & \cdots & -\boldsymbol{M}_{1,c}\boldsymbol{M'}^\top & \boldsymbol{M}_{1,c}\boldsymbol{I}_c \\ \mathbf{0} & \mathbf{0} & -\boldsymbol{M'}^\top & \boldsymbol{I}_c & \ddots & \vdots & -\boldsymbol{M}_{2,1}\boldsymbol{M'}^\top & \boldsymbol{M}_{2,1}\boldsymbol{I}_c & \cdots & -\boldsymbol{M}_{2,c}\boldsymbol{M'}^\top & \boldsymbol{M}_{2,c}\boldsymbol{I}_c \\ \vdots & \ddots & \ddots & \ddots & \ddots & \mathbf{0} & \vdots & \vdots & \cdots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & -\boldsymbol{M'}^\top & \boldsymbol{I}_c & -\boldsymbol{M}_{k,1}\boldsymbol{M'}^\top & \boldsymbol{M}_{k,1}\boldsymbol{I}_c & \cdots & -\boldsymbol{M}_{k,c}\boldsymbol{M'}^\top & \boldsymbol{M}_{k,c}\boldsymbol{I}_c \end{bmatrix}$$

with $c = (n-k)$. In particular, the matrix $\boldsymbol{A}$ has full (row) rank due to the presence of k identity blocks $\boldsymbol{I}_{n-k}$.

Proposition 3. *Given $t > 0$ samples from $\mathcal{L}_{n,k,q,\boldsymbol{Q}}$*

$$(\boldsymbol{G}_i = (\boldsymbol{I}|\boldsymbol{M_i}),\ \boldsymbol{G}'_i = \mathsf{SF}(\boldsymbol{G}_i\boldsymbol{Q}) = (\boldsymbol{I}|\boldsymbol{M'_i}))\,, \qquad i = 1, \dots, t,$$

for a fixed secret $\boldsymbol{Q} \in \mathsf{Mono}_n(\mathbb{F}_q)$, define the following matrix

$$\boldsymbol{A} = \begin{bmatrix} (\,\boldsymbol{I}_k \big| \boldsymbol{M_1}\,) \otimes (\,-\boldsymbol{M}_1'^\top \big| \boldsymbol{I}_{n-k}\,) \\ (\,\boldsymbol{I}_k \big| \boldsymbol{M_2}\,) \otimes (\,-\boldsymbol{M}_2'^\top \big| \boldsymbol{I}_{n-k}\,) \\ \cdots \\ (\,\boldsymbol{I}_k \big| \boldsymbol{M_t}\,) \otimes (\,-\boldsymbol{M}_t'^\top \big| \boldsymbol{I}_{n-k}\,) \end{bmatrix}. \tag{4}$$

Then $\operatorname{rank}(\boldsymbol{A}) < n^2$*.*

Proof. Given that every LCE sample brings $k(n-k)$ rows to the matrix $\boldsymbol{A}$, there are in total $tk(n-k)$ rows and n^2 columns. If $t < \left\lfloor \frac{n^2}{k(n-k)} \right\rfloor + 1$, then the number of rows is smaller than n^2 and the rank cannot reach n^2. Otherwise, there are always more rows than columns, hence the rank of $\boldsymbol{A}$ can be at most n^2. However, by construction we have that $\boldsymbol{A} \cdot \mathsf{vec}(\boldsymbol{Q}) = \mathbf{0}$, hence there exists at least one linear combination of the columns of $\boldsymbol{A}$ that gives the zero vector. It follows that $\boldsymbol{A}$ cannot be full-rank, and so $\operatorname{rank}(\boldsymbol{A}) < n^2$. □

Studying the probability that the rank of $\boldsymbol{A}$ in Equation (4) is maximal is not an easy task. The right kernel of $\boldsymbol{A}$ contains solutions of the form $\mathsf{vec}(\boldsymbol{X})$ such that $\boldsymbol{G}'_i = \mathsf{SF}(\boldsymbol{G}_i\boldsymbol{X})$, for $i = 1, \dots t$, where $\boldsymbol{X}$ is not necessarily monomial. In other words, such a kernel can be written as follows

$$\bigcap_{i=1,\dots,t} \{\mathsf{vec}(\boldsymbol{X}) : \boldsymbol{G}'_i = \mathsf{SF}(\boldsymbol{G}_i\boldsymbol{X})\}\,.$$

For $t > 2$, the inclusion/exclusion principle does not hold in general, and one cannot use it to estimate the dimension of such intersection of vector spaces. Nevertheless, we experimentally studied the probability that the matrix $\boldsymbol{A}$ has maximal rank (i.e. $n^2 - 1$). Based on our experiments, which are reported in Section 3.2, we consider the following assumption.

Assumption 1 *For a given code rate r, there exist positive integers n_0, q_0 such that, for $n > n_0$, $q > q_0$, and $t \geq \left\lfloor \frac{1}{r(1-r)} \right\rfloor + 1 = \left\lfloor \frac{n^2}{k(n-k)} \right\rfloor + 1$, the matrix $\boldsymbol{A}$ constructed from t random samples from $\mathcal{L}_{n,k,q,\boldsymbol{Q}}$ as in Equation (4) has rank equal to $n^2 - 1$ with non-negligible probability in n and q.*

We stress that Assumption 1 is meant to cover all cryptographically interesting cases. Indeed, experimentally, we observed that such an assumption does not hold true either for codes of very short length and small field or for very small rate r, which are not of cryptographic interest. For example, for $r = \frac{1}{2}$, Assumption 1 seems to hold for $n_0 = 8$ and $q_0 = 3$. Under the hypothesis of Assumption 1, we give the sample complexity of LCE in Lemma 1.

Lemma 1. *For $t \geq \left\lfloor \frac{n^2}{k(n-k)} \right\rfloor + 1$ and under Assumption 1, t-LCE is solvable with non-negligible probability in time $O(n^{2\omega})$.*

Proof. Construct the matrix $\boldsymbol{A}$ from t LCE samples sharing the same secret $\boldsymbol{Q}$ as in Proposition 3. Following Assumption 1, the right kernel of $\boldsymbol{A}$ has dimension equal to 1. The generator of such kernel, which can be found via Gaussian elimination, must be (a multiple of) $\mathsf{vec}(\boldsymbol{Q})$ by construction, and so, a solution for each of the t LCE instances. □

Notice that Lemma 1 also applies to PCE as this can be seen as a special case of LCE. We assume an analogue of Assumption 1 for the case of MCE by setting $n := mr$ in the following corollary.

Corollary 1. *For $t \geq \left\lfloor \frac{m^2r^2}{k(mr-k)} \right\rfloor + 1$, t-MCE is solvable with non-negligible probability in time $O((mr)^{2\omega})$.*

In the parameter setting used LESS [2], i.e. $k = n/2$, Lemma 1 says that a constant number of $t = 5$ samples are enough, for any n, to recover the secret monomial. Similarly, for the parameter setting used in MEDS [15], i.e. $k = r = m$, Corollary 1 says that $t = \left\lceil \frac{k^2}{k-1} \right\rceil \approx k$ samples are enough to recover the secret matrix. We stress that this has no implication on the security of such protocols because, in both protocols setting, only one sample is provided.

We verified experimentally in SageMath the correctness of Lemma 1 and Corollary 1, and the scripts are available at [13].

3.1 Implications to ILCE and IMCE

Consider the following ILCE instance

$$\left(\boldsymbol{G},\ \boldsymbol{G}' = \mathsf{SF}(\boldsymbol{G}\boldsymbol{Q}),\ \boldsymbol{G}'' = \mathsf{SF}(\boldsymbol{G}\boldsymbol{Q}^{-1})\right).$$

By multiplying $\boldsymbol{Q}$ to the right in the equation at the right-most entry, one gets

$$(\boldsymbol{G},\ \boldsymbol{G}' = \mathsf{SF}(\boldsymbol{G}\boldsymbol{Q}),\ \boldsymbol{G} = \mathsf{SF}(\boldsymbol{G}''\boldsymbol{Q}))\,,$$

that is *almost* a 2-LCE sample. Indeed, these two resulting LCE samples do not come both from $\mathcal{L}_{n,k,q,\boldsymbol{Q}}$, but are instead related to each other by the matrix $\boldsymbol{G}$ appearing twice, even if on different positions. Nevertheless, we argue below that, for the sake of our analysis, t-ILCE with the above transformation behaves as $2t$-LCE.

Given $t > 0$ random ILCE samples

$$(\boldsymbol{G}_i = (\boldsymbol{I}_k|\boldsymbol{M}_i),\ \boldsymbol{G}'_i = (\boldsymbol{I}_k|\boldsymbol{M}'_i),\ \boldsymbol{G}''_i = (\boldsymbol{I}_k|\boldsymbol{M}''_i)),\quad \text{for } i = 1, \ldots, t,$$

consider the matrix

$$\boldsymbol{A}' = \begin{bmatrix} (\,\boldsymbol{I}_k \big| \boldsymbol{M}_1\,) \otimes (\,-\boldsymbol{M}_1'^{\top} \big| \boldsymbol{I}_{n-k}\,) \\ (\,\boldsymbol{I}_k \big| \boldsymbol{M}_1''\,) \otimes (\,-\boldsymbol{M}_1^{\top} \big| \boldsymbol{I}_{n-k}\,) \\ \cdots \\ (\,\boldsymbol{I}_k \big| \boldsymbol{M}_t\,) \otimes (\,-\boldsymbol{M}_t'^{\top} \big| \boldsymbol{I}_{n-k}\,) \\ (\,\boldsymbol{I}_k \big| \boldsymbol{M}_t''\,) \otimes (\,-\boldsymbol{M}_t^{\top} \big| \boldsymbol{I}_{n-k}\,) \end{bmatrix}. \tag{5}$$

By construction, we have that $\boldsymbol{A}' \cdot \mathsf{vec}(\boldsymbol{Q}) = \boldsymbol{0}$. Hence, for analogous arguments as in the proof of Proposition 3, the matrix $\boldsymbol{A}'$ has rank always smaller than n^2. Looking at matrix $\boldsymbol{A}'$, even if each matrix $\boldsymbol{M}_i$ appears in two row-blocks, they are in different columns and they do not seem to bring any evident linear dependence. In addition, we observed experimentally that, for $t = \left\lfloor \frac{n^2}{2k(n-k)} \right\rfloor + 1$, the probability of rank $(\boldsymbol{A}') \neq n^2 - 1$ is analogous to the case of $2t$-LCE (the results of our experiments are reported in Section 3.2). On the basis of the above considerations, we consider Assumption 2 in order to give the sample complexity of ILCE in Lemma 2.

Assumption 2 *For a given code rate r, there exist positive integers n_0, q_0 such that, for $n > n_0$, $q > q_0$, and $t \geq \left\lfloor \frac{1}{2r(1-r)} \right\rfloor + 1 = \left\lfloor \frac{n^2}{2k(n-k)} \right\rfloor + 1$, the matrix $\boldsymbol{A}'$ constructed from t random samples from $\widehat{\mathcal{L}}_{n,k,q,\boldsymbol{Q}}$ as in Equation (5) has rank equal to $n^2 - 1$ with non-negligible probability in n and q.*

Lemma 2. *Under Assumption 2, for $t \geq \left\lfloor \frac{n^2}{2k(n-k)} \right\rfloor + 1$, t-ILCE is solvable with non-negligible probability in time $O(n^{2\omega})$.*

Proof. Analogous to the proof of Lemma 1. □

Similarly to what is done for MCE, under an analogous assumption to Assumption 2 but for IMCE, we have the following corollary.

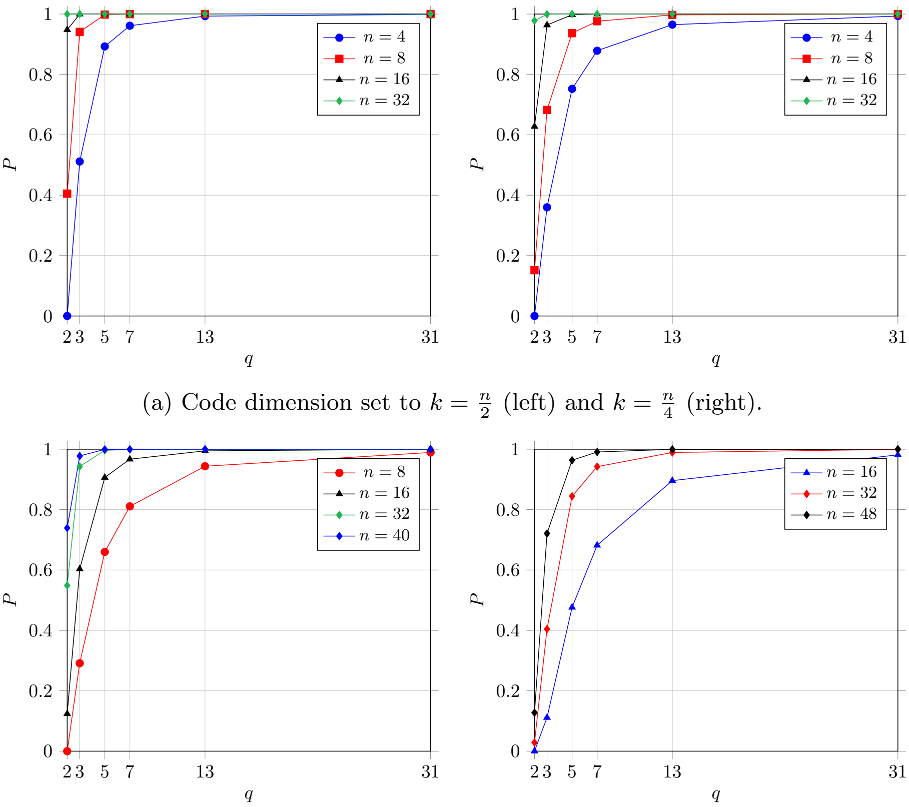
Corollary 2. *For $t \geq \left\lfloor \frac{m^2 r^2}{2k(mr-k)} \right\rfloor + 1$, t-IMCE is solvable with non-negligible probability in time $O((mr)^{2\omega})$.*

Similarly, as above, we verified experimentally in SageMath the correctness of Lemma 2 and Corollary 2.

3.2 Experimental Validation of Assumptions

In this section, we report the results from our experiments for testing whether Assumption 1 and Assumption 2 hold in practice.

Experiments on Assumption 1. Our experiment consists of testing, for a range of n, q and code rate $r = \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}$, how many matrices constructed as in Equation (4) have rank equal to $n^2 - 1$. For each rate, we choose $t = \lfloor \frac{1}{r(1-r)} \rfloor + 1$ and run 10000 trials and report in Figure 1 the fraction of how many trials presented the desired maximal rank. One can see that the measured probability of this event to happen quickly goes to 1 when either or both q and n increase. In addition, one can notice that when the code rate r is close to either 0 or 1, the probability of reaching the maximal rank is lower. Tests for rate $r > \frac{1}{2}$ gave symmetrical results as for $r < \frac{1}{2}$. In overall, our experimental results support Assumption 1.

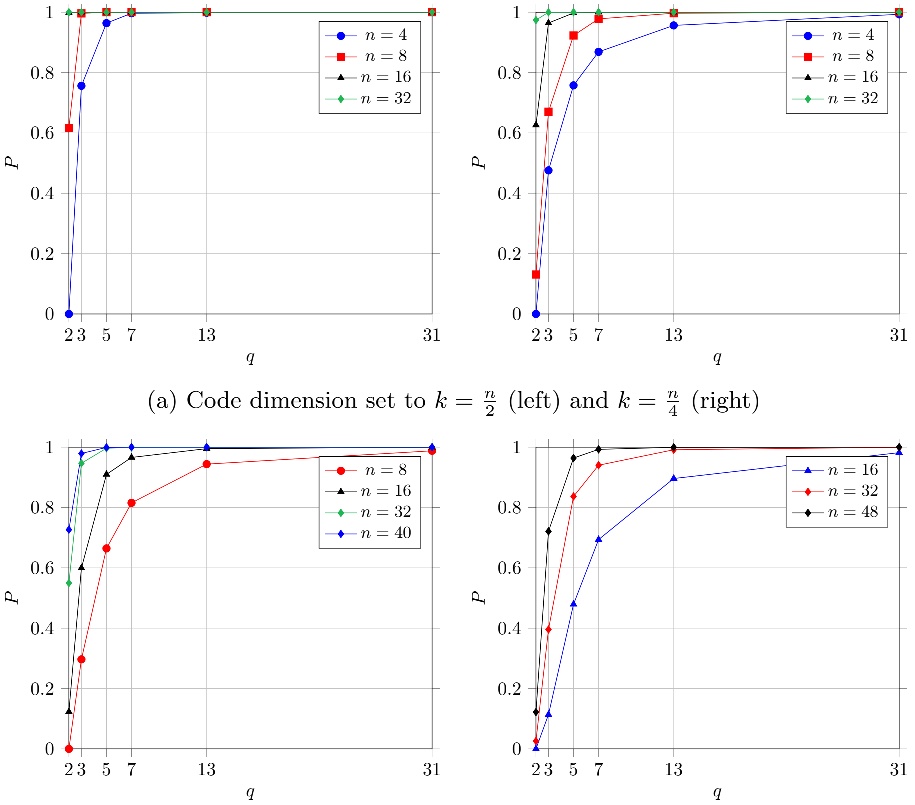


(a) Code dimension set to $k = \frac{n}{2}$ (left) and $k = \frac{n}{4}$ (right).

(b) Code dimension set to $k = \frac{n}{8}$ (left) and $k = \frac{n}{16}$ (right).

Fig. 1: The plots report the measured success rate P over 10000 trials that a matrix $\boldsymbol{A}$ from Equation (4), constructed from $t = \lfloor \frac{n^2}{k(n-k)} \rfloor + 1$ random LCE samples with parameters n, k and q, has rank equal to $n^2 - 1$. Each plots shows the results for different values of n, q and a code rate r equal to $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ and $\frac{1}{16}$.

Experiments on Assumption 2. Under analogous setting of the above experiment, we test whether the matrix $\boldsymbol{A}'$ from Equation (5) has the desired rank n^2-1, for $t = \lfloor \frac{1}{2r(1-r)} \rfloor + 1$. The results reported in Figure 2 support Assumption 2.



(a) Code dimension set to $k = \frac{n}{2}$ (left) and $k = \frac{n}{4}$ (right)

(b) Code dimension set to $k = \frac{n}{8}$ (left) and $k = \frac{n}{16}$ (right)

Fig. 2: The plots report the measured success rate P over 10000 trials that a matrix $\boldsymbol{A}$ from Equation (4), constructed from $t = \lfloor \frac{n^2}{2k(n-k)} \rfloor + 1$ random ILCE samples with parameters n, k and q, has rank equal to n^2-1. Each plots shows the results for different values of n, q and a code rate r equal $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ and $\frac{1}{16}$.

3.3 Solving LCE when rank $(\boldsymbol{A})$ is not maximal

In practice, one can solve t-LCE also when the right rank of the constructed matrix $\boldsymbol{A}$ from Equation (4) is not strictly maximal (i.e. rank $(\boldsymbol{A}) = n^2-1$) as assumed in Assumption 1. Indeed, we show here that if the right kernel of $\boldsymbol{A}$

has dimension between 2 and n, then the monomial solution (or a multiple of it) must be an element of a specific set of generators with high probability, and this set can be computed in polynomial time.

Let us assume that the right kernel of $\boldsymbol{A}$ has dimension n and let $\boldsymbol{g}_1, \dots, \boldsymbol{g}_n \in \mathbb{F}_q^{n^2}$ be its generators (smaller rank cases are analogous). With high probability, there exists a basis transformation that makes $\boldsymbol{g}_1, \dots, \boldsymbol{g}_n$ a standard basis, that is, where

$$\begin{aligned}
\boldsymbol{g}_1 &= (\overbrace{1, 0, \dots, 0}^{n}, *, \dots, *) \\
\boldsymbol{g}_2 &= (0, 1, \dots, 0, *, \dots, *) \\
&\dots \\
\boldsymbol{g}_n &= (0, \dots, 0, 1, *, \dots, *).
\end{aligned}$$

Let us assume that $\mathsf{vec}(\boldsymbol{Q}) \neq \alpha \boldsymbol{g}_i$, for any $\alpha \in \mathbb{F}_q^*$ and $i = 1, \dots, n$. Then it means that $\mathsf{vec}(\boldsymbol{Q})$ must be a linear combination of $\boldsymbol{g}_1, \dots, \boldsymbol{g}_n$. However, because of the monomial structure of $\boldsymbol{Q}$, $\mathsf{vec}(\boldsymbol{Q})$ has only one non-zero entry in its first n entries. It follows that $\mathsf{vec}(\boldsymbol{Q})$ cannot be a combination of two or more $\boldsymbol{g}_i$ as this would generate more than one non-zero entry in these first n positions. Hence, $\mathsf{vec}(\boldsymbol{Q})$ must be one of the elements of the standard basis of the kernel or a multiple of it.

Thanks to this observation, we were able in practice to solve the cases in which $n^2 - n \leq \text{rank}\,(\boldsymbol{A}) \leq n^2 - 1$, and, in particular, for $k = \frac{n}{2}$, we could find the secret monomial with only 4 random LCE samples, which decreases by 1 the sample complexity given by Lemma 1.

4 Further Improvements by Exploiting the Monomial Matrix Structure

In this section, we exploit the structure of the secret matrix in LCE and ILCE to further reduce the number of samples necessary to retrieve the secret monomial. Specifically, we show how to solve, in polynomial time, 2-LCE and ILCE for code rate $\frac{1}{2}$. The approach presented below builds upon the algorithm by Saeed for PCE [33, Sec. 3.7]. As in Section 3, we consider the generators of the codes in systematic form to ease our analysis.

4.1 Solving 2-LCE for $k = n/2$ in polynomial-time

In this section, we introduce a new algorithm that solves 2-LCE in polynomial time for codes of rate $\frac{1}{2}$. Consider a secret matrix $\boldsymbol{Q} \in \mathsf{Mono}_n(\mathbb{F}_q)$ and the following two LCE instances:

$$\begin{aligned}
&\Big(\boldsymbol{G}_1 = (\,\boldsymbol{I}_k | \boldsymbol{M}\,),\, \boldsymbol{G}_1' = \mathsf{SF}(\boldsymbol{G}_1\boldsymbol{Q}) = (\,\boldsymbol{I}_k | \boldsymbol{M}'\,)\Big), \\
&\Big(\boldsymbol{G}_2 = (\,\boldsymbol{I}_k | \boldsymbol{N}\,),\, \boldsymbol{G}_2' = \mathsf{SF}(\boldsymbol{G}_2\boldsymbol{Q}) = (\,\boldsymbol{I}_k | \boldsymbol{N}'\,)\Big).
\end{aligned} \tag{6}$$

for $k = n/2$. We apply Proposition 2 to each instance and write the following homogeneous linear system

$$\mathsf{S}: \quad \overbrace{\begin{bmatrix} (\boldsymbol{I}_k|\boldsymbol{M}) \otimes (-\boldsymbol{M'}^\top|\boldsymbol{I}_{n-k}) \\ (\boldsymbol{I}_k|\boldsymbol{N}) \otimes (-\boldsymbol{N'}^\top|\boldsymbol{I}_{n-k}) \end{bmatrix}}^{\boldsymbol{A}} \mathsf{vec}(\boldsymbol{Q}) = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}. \tag{7}$$

The following proposition gives a sufficient and necessary condition for $\boldsymbol{A}$ to be full-rank.

Proposition 4. *Consider two* LCE *instances as in Equation* (6)*, for* $k = \frac{n}{2}$*. Then the matrix* $\boldsymbol{A}$ *defined in Equation* (7) *is such that* $\mathrm{rank}\,(\boldsymbol{A}) < 2k(n-k)$ *if and only if* $\mathrm{rank}\,(\boldsymbol{M} - \boldsymbol{N}) < k$*.*

Before giving the proof for Proposition 4, we need to prove the following proposition.

Proposition 5. *Under the same setting of Proposition 4, we have that*

$$\mathrm{rank}\,(\boldsymbol{M'} - \boldsymbol{N'}) < k \iff \mathrm{rank}\,(\boldsymbol{M} - \boldsymbol{N}) < k.$$

Proof. First, we prove that $\mathrm{rank}\,(\boldsymbol{M'} - \boldsymbol{N'}) < k \Rightarrow \mathrm{rank}\,(\boldsymbol{M} - \boldsymbol{N}) < k$. One has that

$$\boldsymbol{G}'_1 - \boldsymbol{G}'_2 = (\boldsymbol{I}_k|\boldsymbol{M'}) - (\boldsymbol{I}_k|\boldsymbol{N'}) = \boldsymbol{X}(\boldsymbol{I}_k|\boldsymbol{M})\boldsymbol{Q} - \boldsymbol{Y}(\boldsymbol{I}_k|\boldsymbol{N})\boldsymbol{Q}$$

for some invertible $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{F}_q^{k \times k}$. Then

$$(\boldsymbol{0}|\boldsymbol{M'} - \boldsymbol{N'})\boldsymbol{Q}^{-1} = (\boldsymbol{X}(\boldsymbol{I}_k|\boldsymbol{M}) - \boldsymbol{Y}(\boldsymbol{I}_k|\boldsymbol{N})) = (\boldsymbol{X}|\boldsymbol{Y}) \begin{bmatrix} \boldsymbol{I}_k & \boldsymbol{M} \\ -\boldsymbol{I}_k & -\boldsymbol{N} \end{bmatrix}.$$

For any matrix $\boldsymbol{Z}$ let $\mathsf{ker}_\mathsf{L}(\boldsymbol{Z})$ be its left kernel. Let $\boldsymbol{w}^\top \in \mathsf{ker}_\mathsf{L}(\boldsymbol{M'} - \boldsymbol{N'})$, then

$$\boldsymbol{0} = \boldsymbol{w}^\top \cdot (\boldsymbol{0}|\boldsymbol{M'} - \boldsymbol{N'})\boldsymbol{Q}^{-1} = \boldsymbol{w}^\top \cdot (\boldsymbol{X}|\boldsymbol{Y}) \begin{bmatrix} \boldsymbol{I}_k & \boldsymbol{M} \\ -\boldsymbol{I}_k & -\boldsymbol{N} \end{bmatrix}.$$

It follows that ${\boldsymbol{w}'}^\top = \boldsymbol{w}^\top \cdot (\boldsymbol{X}|\boldsymbol{Y}) \in \mathsf{ker}_\mathsf{L}\left(\begin{bmatrix} \boldsymbol{I}_k & \boldsymbol{M} \\ -\boldsymbol{I}_k & -\boldsymbol{N} \end{bmatrix}\right)$ Notice that $\boldsymbol{w}' \in \mathbb{F}_q^n$ must be of the form $\boldsymbol{w}' = (\boldsymbol{v}, \boldsymbol{v})$, with $\boldsymbol{v} \neq 0 \in \mathbb{F}_q^k$ and $\boldsymbol{v}^\top \in \mathsf{ker}_\mathsf{L}\,(\boldsymbol{M} - \boldsymbol{N})$. Hence we have that $\mathrm{rank}\,(\boldsymbol{M} - \boldsymbol{N}) < k$. The other implication $\mathrm{rank}\,(\boldsymbol{M} - \boldsymbol{N}) < k \Rightarrow \mathrm{rank}\,(\boldsymbol{M'} - \boldsymbol{N'}) < k$ follows by using analogous arguments as above. □

We can now give the proof of Proposition 4.

Proof (Proposition 4). First, we prove that $\mathrm{rank}\,(\boldsymbol{M} - \boldsymbol{N}) < k \Rightarrow \mathrm{rank}\,(\boldsymbol{A}) < 2k(n-k)$. Notice that since $\boldsymbol{A}$ has n^2 columns and $2k(n-k) = \frac{n^2}{2}$ rows, $\mathrm{rank}\,(\boldsymbol{A})$ can be at most equal to $2k(n-k)$. Let $\boldsymbol{v}^\top \neq 0 \in \mathsf{ker}_\mathsf{L}(\boldsymbol{M} - \boldsymbol{N})$, then there exists $\boldsymbol{w}^\top \neq 0 \in \mathsf{ker}_\mathsf{L}(\boldsymbol{M'} - \boldsymbol{N'})$ from Proposition 5. Then we have that

$$(\boldsymbol{v}^\top| - \boldsymbol{v}^\top) \begin{bmatrix} (\boldsymbol{I}_k|\boldsymbol{M}) \\ (\boldsymbol{I}_k|\boldsymbol{N}) \end{bmatrix} = \boldsymbol{0} \quad \text{and} \quad (\boldsymbol{w}^\top| - \boldsymbol{w}^\top) \begin{bmatrix} (-\boldsymbol{M'}^\top|\boldsymbol{I}_{n-k}) \\ (-\boldsymbol{N'}^\top|\boldsymbol{I}_{n-k}) \end{bmatrix} = \boldsymbol{0}.$$

It follows that

$$(\boldsymbol{v}^\top \otimes \boldsymbol{w}^\top | - \boldsymbol{v}^\top \otimes \boldsymbol{w}^\top)\begin{bmatrix}(\boldsymbol{I}_k|\boldsymbol{M}) \otimes (-\boldsymbol{M'}^\top|\boldsymbol{I}_{n-k})\\ (\boldsymbol{I}_k|\boldsymbol{N}) \otimes (-\boldsymbol{N'}^\top|\boldsymbol{I}_{n-k})\end{bmatrix} = \mathbf{0}.$$

Hence, $(\boldsymbol{v}^\top \otimes \boldsymbol{w}^\top | - \boldsymbol{v}^\top \otimes \boldsymbol{w}^\top) \neq 0 \in \mathsf{ker}_{\mathsf{L}}(\boldsymbol{A})$ and so $\text{rank}\,(A) < 2k(n-k)$.

We prove now that $\text{rank}\,(\boldsymbol{A}) < 2k(n-k) \Rightarrow \text{rank}\,(\boldsymbol{M}-\boldsymbol{N}) < k$. Let $\boldsymbol{s}^\top = (\boldsymbol{s}_1{}^\top, \boldsymbol{s}_2{}^\top) \neq 0 \in \mathsf{ker}_{\mathsf{L}}(\boldsymbol{A})$. If we restrict the multiplication $\boldsymbol{s}^\top\boldsymbol{A} = \mathbf{0}$ to the first $2k$ columns of $\boldsymbol{A}$, we get the following equation

$$(\boldsymbol{s}_1^\top|\boldsymbol{s}_2^\top)\begin{bmatrix}-\boldsymbol{M'}^\top & \boldsymbol{I}_{n-k}\\ \mathbf{0} & \mathbf{0}\\ -\boldsymbol{N'}^\top & \boldsymbol{I}_{n-k}\\ \mathbf{0} & \mathbf{0}\end{bmatrix} = \mathbf{0}.$$

Let $\bar{\boldsymbol{s}}_1, \bar{\boldsymbol{s}}_2 \in \mathbb{F}_q^k$ be the vectors of the first k entries of $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ respectively. Then we have that

$$(\bar{\boldsymbol{s}}_1^\top|\bar{\boldsymbol{s}}_2^\top)\begin{bmatrix}-\boldsymbol{M'}^\top & \boldsymbol{I}_{n-k}\\ -\boldsymbol{N'}^\top & \boldsymbol{I}_{n-k}\end{bmatrix} = \mathbf{0}.$$

It follows that $\bar{\boldsymbol{s}}_1^\top + \bar{\boldsymbol{s}}_2^\top = 0$, therefore $-\bar{\boldsymbol{s}}_1^\top = \bar{\boldsymbol{s}}_2^\top$ and that $\bar{\boldsymbol{s}}_1^\top \in \mathsf{ker}_{\mathsf{L}}(\boldsymbol{M'}^\top - \boldsymbol{N'}^\top)$ and, for $k = \frac{n}{2}$, $\text{rank}\,(\boldsymbol{M'} - \boldsymbol{N'}) < k$. From Proposition 5 we conclude that $\text{rank}\,(\boldsymbol{M} - \boldsymbol{N}) < k$. □

Description of the Algorithm. The main idea of our algorithm is to infer information about the secret monomial matrix $\boldsymbol{Q}$ by guessing the position of the non-zero entry in each row and checking whether the resulting reduced system admits solutions. More specifically, we iteratively guess the entries of $\boldsymbol{Q}$ to be non-zero. Each guess consists of evaluating the variable corresponding to the (i,j)-th entry of $\boldsymbol{Q}$ to be equal to 1. Thanks to the monomial structure of $\boldsymbol{Q}$, this results in guessing a total of $2n-1$ variables simultaneously, since all the remaining variables in the i-th row and in the j-th column must be equal to 0. Then, we either retain or discard the guess depending on whether the reduced linear system obtained from such a guess admits any solution(s) or not.

We now explain why such a guess on the correct non-zero position of $\boldsymbol{Q}$ is still useful even if $\boldsymbol{Q}(i,j) \neq 1$. Recall that $\boldsymbol{Q} = \boldsymbol{P}\boldsymbol{D}$, where $\boldsymbol{P}$ is a permutation matrix in $\mathsf{Perm}_n(\mathbb{F}_q)$ and $\boldsymbol{D}$ is a diagonal matrix in $\mathsf{GL}_n(\mathbb{F}_q)$. Let $d_i \in \mathbb{F}_q^*$ be the i-th diagonal entry of $\boldsymbol{D}$. Then $\boldsymbol{R}_i = d_i^{-1}\boldsymbol{Q}$ satisfies $\boldsymbol{G}_1' = \mathsf{SF}(\boldsymbol{G}_1\boldsymbol{R}_i)$ and $\boldsymbol{G}_2' = \mathsf{SF}(\boldsymbol{G}_2\boldsymbol{R}_i)$ for each $i \in \{1,\ldots,n\}$. In other words, this guess restricts the set of possible solutions to include a specific multiple $\boldsymbol{R}_i$ of $\boldsymbol{Q}$ that has 1 in its i-th non-zero entry (due to scaling by d_i) which also serves as a solution to the given 2-LCE instance. Therefore, such an evaluation on the non-zero entry remains valid.

We give here a characterization of the linear system obtained by guessing a single position. Setting $\boldsymbol{Q}(i,j) = 1$ and $\boldsymbol{Q}(i,\mu), \boldsymbol{Q}(\eta,j) = 0$, for $\mu \in \{1\ldots n\}\backslash\{j\}$

and $\eta \in \{1 \dots n\} \setminus \{i\}$, results in removing the corresponding $2n-1$ columns of $\boldsymbol{A}$ from the linear system S in Equation (7). This operation produces the linear system

$$\mathsf{S}_{i,j} : \boldsymbol{A}_{ij} \cdot \mathsf{vec}(\boldsymbol{Q}') = \boldsymbol{b}_{ij} \tag{8}$$

in dimension $2k(n-k) \times (n-1)^2$, where

$$\begin{aligned} \boldsymbol{A}_{ij} &= \begin{bmatrix} (\,\boldsymbol{I}_k \big| \boldsymbol{M}\,)_{-i} \otimes (\,-\boldsymbol{M}'^{\top} \big| \boldsymbol{I}_{n-k}\,)_{-j} \\ (\,\boldsymbol{I}_k \big| \boldsymbol{N}\,)_{-i} \otimes (\,-\boldsymbol{N}'^{\top} \big| \boldsymbol{I}_{n-k}\,)_{-j} \end{bmatrix}, \\ -\boldsymbol{b}_{ij} &= \begin{bmatrix} (\,\boldsymbol{I}_k \big| \boldsymbol{M}\,)_{i} \otimes (\,-\boldsymbol{M}'^{\top} \big| \boldsymbol{I}_{n-k}\,)_{j} \\ (\,\boldsymbol{I}_k \big| \boldsymbol{N}\,)_{i} \otimes (\,-\boldsymbol{N}'^{\top} \big| \boldsymbol{I}_{n-k}\,)_{j} \end{bmatrix}, \end{aligned}$$

and $\boldsymbol{Q}'$ is the $(n-1) \times (n-1)$ matrix obtained by removing the i-th row and j-th column from $\boldsymbol{Q}$. In other words, we obtain a new non-homogeneous linear system given by the tensor product of $\boldsymbol{G}$ punctured at position i and $\boldsymbol{H}'$ (parity check matrix of $\boldsymbol{G}'$) punctured at position j. Notice that the vector of the constant terms $\boldsymbol{b}_{ij}$ corresponds to the $(n(i-1)+j)$-th column of the original matrix $\boldsymbol{A}$, i.e., the one corresponding to the variable $\boldsymbol{Q}(i,j)$ that is guessed to be non-zero.

On each guess, we use the following test to accept or reject a guess.

Test 1 *For the guess on the (i,j)-th entry of $\boldsymbol{Q}$ to be non-zero, construct a reduced system S_{ij} from S (as in Equation (8)) with $(n-1)^2$ variables by setting $\boldsymbol{Q}(i,j) = 1$ and $\boldsymbol{Q}(i,\mu), \boldsymbol{Q}(\eta,j) = 0$, for $\mu \in \{1 \dots n\} \setminus \{j\}$ and $\eta \in \{1 \dots n\} \setminus \{i\}$. Accept the guess if the system S_{ij} accepts at least one solution, reject otherwise.*

We use Rouché–Capelli Theorem to check whether S_{ij} accepts solutions or not. Indeed, the system S_{ij} accepts solutions if and only if $\operatorname{rank}(\boldsymbol{A}_{ij}) = \operatorname{rank}(\boldsymbol{A}_{ij}|\boldsymbol{b}_{ij})$. When a guess is rejected, this means that no solution in S exists with $\boldsymbol{Q}(i,j) \neq 0$. Hence, the variable corresponding to $\boldsymbol{Q}(i,j)$ in S is set to zero. If enough variables are set to zero after the guessing procedure, i.e. the system becomes (over)determined, and we can retrieve the remaining ones using Gaussian elimination. The whole strategy is outlined in Algorithm 1.

Algorithm 1 Solving 2-LCE

```
Input: A 2-LCE instance as in Equation (6)
Output: A monomial matrix $\boldsymbol{R}$, solution to Equation (6) or $\perp$
Construct the linear system S given by Equation (7)
Set $g = [g_1, \dots, g_n]$ such that $g_i$ is an empty list
for $i := 1$ to $n$ do                                          ▷ loop over rows
    for $j := 1$ to $n$ do                                      ▷ loop over columns
        if Test 1 passes then
            Append $j$ to the list $g_i$
        end if
    end for
end for
Construct the linear system $\mathsf{S}_{\mathsf{red}}$ obtained by substituting $\boldsymbol{Q}(i, j) = 0$ in S for each
    $i := 1, \dots, n$ and $j \notin \mathsf{g}_i$
if $\mathsf{S}_{\mathsf{red}}$ is underdetermined then
    Return $\perp$
end if
Compute a solution matrix $\boldsymbol{R}$ of the linear system $\mathsf{S}_{\mathsf{red}}$
Return $\boldsymbol{R}$
```

Notice that, when Algorithm 1 succeeds, it returns an equivalent solution (a scalar multiple of) to the original secret matrix $\boldsymbol{Q}$.

Heuristic analysis of Algorithm 1. First of all, notice that Test 1 always accepts a correct guess since, in this case, S_{ij} accepts solutions by construction. On the other hand, Test 1 may or may not accept a wrong guess. Thus, we begin our analysis by estimating the probability that Test 1 accepts a wrong guess.

Proposition 4 gives the condition for which the matrix of coefficients $\boldsymbol{A}$ in Equation (7) is full rank. In particular, given that $\boldsymbol{M}, \boldsymbol{N}$ are sampled uniformly at random, we know that

$$Pr\left(\operatorname{rank}(\boldsymbol{A}) = \frac{n^2}{2}\right) = Pr\left(\operatorname{rank}(\boldsymbol{M} - \boldsymbol{N}) = k\right) = 1 - \frac{1}{q}. \tag{9}$$

On the other hand, Proposition 1 applied to the matrix of coefficients $\boldsymbol{A}_{ij}$ of the reduced system S_{ij} (Equation (8)) tells us that $\operatorname{rank}(\boldsymbol{A}_{ij}) = \frac{n^2}{2} - d$, for some $d > 0$. Using Rouché–Capelli Theorem to check whether S_{ij} admits solutions or not, implies that the guess (i, j) passes Test 1 if and only if $\operatorname{rank}(\boldsymbol{A}_{ij}|\boldsymbol{b}_{ij}) = \operatorname{rank}(\boldsymbol{A}_{ij}) = \frac{n^2}{2} - d$.

Let X be the left kernel of $\boldsymbol{A}_{ij}$. The dimension of X is $\frac{n^2}{2} - \operatorname{rank}(\boldsymbol{A}_{ij}) = d$, and let $\boldsymbol{B}_X \in \mathbb{F}_q^{d \times \frac{n^2}{2}}$ be its generator matrix. Similarly, Let Y be the left kernel of $\boldsymbol{b}_{ij}$ of dimension $\frac{n^2}{2} - \operatorname{rank}(\boldsymbol{b}_{ij}) = \frac{n^2}{2} - 1$ and let $\boldsymbol{B}_Y \in \mathbb{F}_q^{(\frac{n^2}{2}-1) \times \frac{n^2}{2}}$ be its generator matrix. Then, we have that

$$\operatorname{rank}(\boldsymbol{A}_{ij}|\boldsymbol{b}_{ij}) = \operatorname{rank}(\boldsymbol{A}_{ij}) \iff X \subset Y \iff \operatorname{rank}(\boldsymbol{B}_Y) = \operatorname{rank}\left(\begin{bmatrix}\boldsymbol{B}_Y \\ \boldsymbol{B}_X\end{bmatrix}\right).$$

Heuristically, we model $\boldsymbol{B}_X$ and $\boldsymbol{B}_Y$ as random matrices, and the probability that all rows of $\boldsymbol{B}_X$ are linearly dependant from the rows of $\boldsymbol{B}_Y$ is approximately equal to $\frac{1}{q^d}$. Therefore, the expected probability that a wrong guess passes Test 1 is $\frac{1}{q^d}$.

Let us now estimate the expected number of variables that will pass Test 1, i.e., the number of variables of the system $\mathsf{S}_{\mathsf{red}}$ in Algorithm 1. Here, we consider the most probable scenario of $d = 1$ (that is also the worst case scenario, since for $d > 1$ Test 1 accepts wrong guesses with lower probability). In total, there are n correct guesses (one for each row) that will always pass Test 1, and the remaining $n^2 - n$ incorrect guesses will pass with probability $\frac{1}{q}$. The expected number of survival variables is

$$N = n + (n^2 - n)\frac{1}{q}. \tag{10}$$

We have that the resulting system is (over)determined when $N \leq \frac{n^2}{2}$, and this is true when

$$q \geq \frac{2(n-1)}{n-2}. \tag{11}$$

Notice that, for $q > 2$ and $n \geq 4$, Equation (11) is always satisfied. Hence, when the parameters satisfy Equation (11), the condition that determines the success of Algorithm 1 is that $\operatorname{rank}(\boldsymbol{A}) = \frac{n^2}{2}$, which happens with probability $1 - \frac{1}{q}$ (see Equation (9)).

Complexity. The computational cost of checking $\operatorname{rank}(\boldsymbol{A}_{ij}|\boldsymbol{b}_{ij}) = \operatorname{rank}(\boldsymbol{A}_{ij})$ is $O(n^{2\omega})$, for $\omega \in [2, 3]$. This computation must be repeated for n^2 guesses, giving a computational complexity of

$$O(n^{2+2\omega})$$

field operations. The memory complexity is of $O(n^4)$ field elements.

In order to check the correctness of Algorithm 1 and of the proposed analysis, we perform extensive experiments in SageMath up to code length $n = 128$ as discussed in Section 5.1.

4.2 Solving ILCE for $k = n/2$ in polynomial-time

Consider an ILCE instance

$$(\boldsymbol{G} = (\boldsymbol{I}_k|\boldsymbol{M}),\ \boldsymbol{G}' = (\boldsymbol{I}_k|\boldsymbol{M}'),\ \boldsymbol{G}'' = (\boldsymbol{I}_k|\boldsymbol{M}'')).$$

Following the same reasoning as in Section 3.1, we obtain a system which is *almost* same as the one obtained from a 2-LCE instance. For Algorithm 1 to work, we need first to check that the following matrix

$$\boldsymbol{A}' = \begin{bmatrix} (\boldsymbol{I}_k|\boldsymbol{M}) \otimes (-\boldsymbol{M}'^\top|\boldsymbol{I}_{n-k}) \\ (\boldsymbol{I}_k|\boldsymbol{M}'') \otimes (-\boldsymbol{M}^\top|\boldsymbol{I}_{n-k}) \end{bmatrix}. \tag{12}$$

is full rank. According to Proposition 4, $\boldsymbol{A}'$ is full-rank if and only if $\mathsf{ker}_{\mathsf{L}}\left(\boldsymbol{M}-\boldsymbol{M}''\right)$ is trivial. Heuristically, given that $\boldsymbol{M}$ is random and modelling $\boldsymbol{M}''$ as also random, $\boldsymbol{A}'$ is full rank with probability $1-\frac{1}{q}$. Then, guessing variables of the system $\boldsymbol{A}' \cdot \mathsf{vec}(\boldsymbol{Q}) = \boldsymbol{0}$ produces a system analogous to Equation (8), where Proposition 1 naturally applies to $\boldsymbol{A}'$. Consequently, the requirements for Algorithm 1 are met, allowing us to solve ILCE in polynomial time.

Our experiments, reported in Section 5.2, show that Algorithm 1 solves 2-LCE and ILCE with analogous success probability.

4.3 Solving 2-PCE and IPCE for self-dual codes

Since PCE is a special case of LCE, the above results also apply to IPCE and 2-PCE for random codes. However, in this case, it is already known that PCE can be solved in polynomial time using the Support Splitting Algorithm [36]. Unfortunately, for the case of self-dual codes, this approach has exponential time complexity.

We argue that the hull of the code does not play a role in our algorithm. First, notice that when building the system in Equation (7), the code and its dual are never used simultaneously (instead, we use the dual of an equivalent code). Specifically, given two PCE instances

$$(\boldsymbol{G}_1, \boldsymbol{G}_1') \qquad (\boldsymbol{G}_2, \boldsymbol{G}_2'),$$

with secret permutation matrix $\boldsymbol{P}$, we construct the system (notice that in this case, $\boldsymbol{G}_i'$ is the dual of itself)

$$\begin{bmatrix} \boldsymbol{G}_1 \otimes \boldsymbol{G}_1' \\ \boldsymbol{G}_2 \otimes \boldsymbol{G}_2' \end{bmatrix} \mathsf{vec}(\boldsymbol{P}) = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}. \tag{13}$$

Second, the key factor that makes the Rouché–Capelli test work is that the matrix of coefficients after puncturing $\boldsymbol{A}_{ij}$ must not be full-rank, and this is true via Proposition 1 regardless of the dimension of the hull. Finally, our experiments, reported in Section 5.3, show that our algorithm solves both IPCE and 2-PCE for self-dual code instances similarly to the case of random code instances (with trivial hull).

4.4 Comparisons with Saeed's algorithm [33]

In [33, Section 3.7], Saeed proposed an algorithm to solve PCE for random code instances. Let the following

$$\left(\boldsymbol{G}_1 = (\,\boldsymbol{I}_k | \boldsymbol{M}\,), \boldsymbol{G}_2 = \mathsf{SF}(\boldsymbol{G}_1 \boldsymbol{P}) = (\,\boldsymbol{I}_k | \boldsymbol{M}'\,)\right),$$

be a PCE instance, where $\boldsymbol{P} \in \mathsf{Perm}_n(\mathbb{F}_q)$. From this only sample, they construct the following linear system

$$\begin{bmatrix} (\,\boldsymbol{I}_k|\boldsymbol{M}\,) \otimes (\,-\boldsymbol{M}'^{\top}|\boldsymbol{I}_{n-k}\,) \\ (\,-\boldsymbol{M}^{\top}|\boldsymbol{I}_{n-k}\,) \otimes (\,\boldsymbol{I}_k|\boldsymbol{M}'\,) \\ \boldsymbol{I}_n \otimes \boldsymbol{1}_n^{\top} \\ \boldsymbol{1}_n^{\top} \otimes \boldsymbol{I}_n \end{bmatrix} \mathsf{vec}(\boldsymbol{P}) = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{1}_n \\ \boldsymbol{1}_n \end{bmatrix}, \tag{14}$$

where $\boldsymbol{1}_n$ is the column vector of length n and 1 in each entry. Notice that the first equations block is analogous to the one of Equation (7). The second block is obtained thanks to the following observation: since $\boldsymbol{P}^{-1} = \boldsymbol{P}^{\top}$, we have that $\boldsymbol{G}_1 = \mathsf{SF}(\boldsymbol{G}_2\boldsymbol{P}^{-1}) = \mathsf{SF}(\boldsymbol{G}_2\boldsymbol{P}^{\top})$ also holds. However, these new equations are, in general, linearly independent from the above only when the hull of the code is trivial. The last two equation blocks simply condition the sum of the elements of $\boldsymbol{P}$ in the same row and column to equal 1, which is true for every permutation matrix.

Starting from the system in Equation (14), Saeed's algorithm works similarly to ours. However, our algorithm proposes a more efficient method for recovering the final secret (Line 14 of Algorithm 1). Specifically, our heuristic analysis shows that the number of survival variables is smaller than or equal to the number of equations, allowing an efficient recovery of the secret via Gaussian elimination. In contrast, the author of [33] does not present such an analysis, and they also do not specify how to recover the final solution. They, in fact, speculate that retrieving the solution may be computationally expensive as this step may require an exhaustive search on a large set.[7]

5 Experiments

We support the findings presented in this manuscript with extensive experiments and simulations performed by means of a SageMath [38] proof-of-concept implementation available at [13]. Regarding Section 3, we provide the scripts to test the correctness of Lemmas 1 and 2 and Corollaries 1 and 2. For Section 4, we report in this section the results of extensive experiments performed on solving 2-LCE/ILCE with random codes, and 2-PCE/IPCE with self-dual codes.

5.1 Solving 2-LCE

We perform extensive experiments to corroborate the weaker security provided by 2-LCE and ILCE when compared to LCE. We take into consideration the following observation on the parameter set from [2]:

- 128 bits: $n = 252$ and $q = 127$ satisfies $q \approx n/2$,
- 192 bits: $n = 400$ and $q = 127$ satisfies $q \approx n/3$,
- 256 bits: $n = 548$ and $q = 127$ satisfies $q \approx n/4$.

[7] In [33, page 62], the author says "*This might have high complexity depending on the size of the solution set.*" We interpret this as requiring an exhaustive search.

To the best of our knowledge, the concrete security of 2-LCE was not analyzed before this work, and therefore we test our results on 2-LCE using the parameters providing different security levels for LCE. Thus, we focus on the following parameter set: $n \in [32, 40, 48, 64, 72, 80, 96, 128]$, $k = n/2$, and $q \in [n/2, n/3, n/4, 127]$. Essentially, we tackle cases that are believed to provide security equivalent to $20-70$ bits in the case of LCE; such a complexity estimation is based on the analysis presented in [2]. Table 3 presents our experiments' time and memory measurements on a 2.45 GHz AMD EPYC 7763 64-core Processor machine with 1T of RAM running Ubuntu 22.04.2 LTS.

From some preliminary experiments, we observed that the upper bound in Proposition 1 is reached with overwhelming probability, i.e., $\mathrm{rank}\,(\boldsymbol{A}_{ij}) = \frac{n^2}{2} - 1$. Hence, we optimize the algorithm and avoid one rank computation per guess by substituting the Rouché–Capelli test with the test of checking whether $\mathrm{rank}\,(\boldsymbol{A}_{ij}|\boldsymbol{b}_{ij}) = \frac{n^2}{2} - 1$ or not.

Our implementation employs parallelization per row; more precisely, it runs n processors in parallel, and the j^{th} processor has the task of computing the rank of $\mathsf{S}_{i,j}$. Consequently, that parallelization approach gives a factor of n times faster, but the memory increases by the same factor (i.e., it is n times bigger). We use the multiprocessing Python package for the parallelization and the tracemalloc Python module to measure the memory usage. In addition, for each parameter set considered, Table 3 reports a comparison of the expected number of variables in $\mathsf{S}_{\mathsf{red}}$ against the average obtained in our experiments. This comparison illustrates that our experimental findings align with the analysis presented in Section 4.

5.2 Solving ILCE

We report the results of the experiments that we performed to support our claims in Section 4.2, i.e., Algorithm 1 solves ILCE analogously to 2-LCE. For different values of n and q, we report in Table 4 the measured success rate over 100 trials of both problems. One can see that 2-LCE and ILCE get solved with approximately the same success probability and that this corresponds to the success condition probability of Algorithm 1 (Equation (9)).

5.3 Solving Self Dual 2-PCE and IPCE Instances

In this section, we report the results of our experiments to support our claim in Section 4.3, that is, Algorithm 1 solves 2-PCE and IPCE with self-dual codes instances analogously to random codes instances.

We consider the set of self-dual codes generators provided in [20,21], for $n \in \{16, 24, 28, 36, 40, 44\}$, $k = n/2$, and $q = 7$. Given that, for each n only one generator $\boldsymbol{G}$ is given, we compute different 2-PCE instances at every test iteration as follows. First, we compute the generator of an equivalent code $\boldsymbol{G}_1$ of $\boldsymbol{G}$ through a random permutation $\boldsymbol{T}$, and then we compute a PCE instance as $(\boldsymbol{G}_1, \boldsymbol{G}_2 = \mathsf{SF}(\boldsymbol{G}_1\boldsymbol{P}))$, where $\boldsymbol{P}$ is the random secret permutation to discover. Table 5 reports the success rate over 100 trials, for the available values of n.

n	q	Estimated LCE bit security	Expected vars in $\mathsf{S}_{\mathsf{red}}$	Measured vars in $\mathsf{S}_{\mathsf{red}}$	Memory (GB)	Runtime	Ratio
32	7	20	178	178	1.03	20s	18/20
	11	22	125	124	1.02	19s	14/20
	17	23	92	93	1.03	19s	19/20
	127	29	40	40	1.05	19s	20/20
40	11	25	185	183	2.57	48s	20/20
	13	25	163	165	2.56	47s	20/20
	19	27	124	121	2.56	47s	19/20
	127	33	53	54	2.57	47s	19/20
48	13	28	225	231	5.34	01m 41s	19/20
	17	29	183	173	5.35	01m 44s	18/20
	23	31	148	146	5.34	01m 44s	19/20
	127	37	66	69	5.36	01m 43s	20/20
64	17	35	305	288	16.96	07m 08s	17/20
	23	37	242	240	16.96	07m 00s	17/20
	31	38	196	191	16.96	07m 06s	20/20
	127	44	96	97	16.97	07m 02s	20/20
72	19	39	345	343	27.19	13m 27s	20/20
	23	40	297	291	27.19	13m 58s	17/20
	37	42	212	212	27.20	12m 50s	18/20
	127	47	113	113	27.21	13m 08s	20/20
80	19	41	416	417	41.48	21m 40s	18/20
	29	44	301	302	41.50	21m 48s	20/20
	41	46	236	228	41.49	18m 37s	18/20
	127	51	130	132	41.50	18m 09s	20/20
96	23	48	496	499	86.10	01h 04m	20/20
	31	51	393	392	86.10	01h 04m	19/20
	47	54	292	284	86.10	01h 04m	20/20
	127	58	169	169	86.09	01h 08m	20/20
128	31	63	656	639	272.06	06h 02m	20/20
	43	66	509	519	272.07	06h 02m	19/20
	61	69	397	397	272.06	05h 51m	19/20
	127	73	257	252	272.10	04h 39m	20/20

Table 3: The data corresponds to the average of solving 20 random 2-LCE instances. The fourth and the fifth columns present the expected number of variables in $\mathsf{S}_{\mathsf{red}}$ according to Equation (10) and the average of the observed values, respectively. The last column presents the number of successfully solved random 2-LCE instances (i.e., the success ratio obtained from the experiments).

One can note that our algorithm succeeds with probability approximately equal

q \ n		16	24	32	40	$1 - \frac{1}{q}$
7	2-LCE	0.81	0.84	0.81	0.86	0.86
	ILCE	0.87	0.82	0.86	0.85	
11	2-LCE	0.92	0.87	0.93	0.87	0.91
	ILCE	0.91	0.93	0.89	0.90	
17	2-LCE	0.95	0.95	0.93	0.92	0.94
	ILCE	0.96	0.94	0.96	0.96	
31	2-LCE	0.96	0.99	0.96	0.95	0.97
	ILCE	0.94	0.96	0.98	0.98	

Table 4: The data corresponds to the number of solved instances divided by the total number of experiments (which is 100). The last column reports the expected success probability from our analysis, that is, the system in Equation (7) is full-rank. In all the experiments, we have $k = n/2$.

to $1 - \frac{1}{q} \approx 0.86$, matching the probability of our success condition, that is, the coefficient matrix in Equation (13) is full-rank (see Equation (9)).

n	16	24	28	36	40	44
2-PCE	0.85	0.86	0.88	0.91	0.87	0.89
IPCE	0.85	0.83	0.88	0.84	0.90	0.86

Table 5: The data corresponds to the number of solved self-dual instances divided by the total number of experiments (which is 100). In all the experiments, we have $q = 7$ and $k = n/2$.

6 Cryptographic implications

To better illustrate the impact of the results from Section 4, we start by giving a comparison between the estimated complexities of LCE according to [2], and the complexity for 2-LCE and ILCE according to Section 4. We follow the parameter sets from [2], ensuring 128, 192, and 256 security bits for LCE under the current most efficient algorithms for solving it. On the other hand, the estimations from Section 4 imply a security of 2-LCE and ILCE of around 60-70 security bits for the same parameter sets (see Table 6).

n	k	q	LCE	2-LCE & ILCE
252	126	127	128	61
400	200	127	192	66
548	274	127	256	70

Table 6: The column corresponding to LCE is according to the security analysis from [2]. The column corresponding to 2-LCE & ILCE concerns the complexity of Algorithm 1 (detailed in Section 4) with $\omega = \log_2(7)$. The presented numbers are given in logarithm base two.

On the impact on ILCE*-based linkable signatures:* In [4], the authors stated that if the ILCE problem were proved to be safe, all the necessary linkable properties would be satisfied, thus building a secure linkable ring signature scheme. Nevertheless, as a direct consequence of Section 4.2, we have that any linkable signature relying on the hardness of the ILCE problem is insecure when the conditions from Section 4 are satisfied.

On the impact on 2-LCE*-based threshold signatures:* The authors of [7] introduced the 2-LCE problem in the group action framework [7, Problem 3] and emphasized constructions for 2-weakly pseudorandom scenarios. Specifically, they proposed a threshold signature whose distributed key generation algorithm is based on the conjectured 2-weakly pseudorandom group actions built on top of the LCE and MCE problems. Nevertheless, as another consequence of Section 4, we show that Definition 7 when instantiated with group action based on LCE does not achieve the pseudorandomness property as we can use Algorithm 1 to recover the secret, which breaks the unpredictability as well as the pseudorandomness of the group action. Therefore, the threshold signature instantiations with LESS from [7, Sec. 5.3] become insecure when $k = n/2$.[8]

Other implications: D'Alconzo and Di Scala have demonstrated that the LCE and MCE group actions do not guarantee weak unpredictability and weak pseudorandomness properties [19]. However, their findings do not apply when the instances are given in systematic form. Our work addresses this gap by providing a more general framework that includes the systematic form case. In light of this, Table 1 summarizes the primitives that, with instantiations from the literature, can and cannot be constructed using these group actions.

What about the implications to 2-MCE*?* Given that the secret matrices of 2-MCE instances do not have the monomial structure, the algorithms from Section 4 do not apply to solving 2-MCE instances. In particular, it is unclear how to perform a similar guessing on the entries of the secret matrices.

[8] The authors published an updated version of their protocol that does not rely on 2-LCE as a preprint after our attack was made public [6].

Acknowledgments

Giuseppe D'Alconzo and Antonio J. Di Scala are members of GNSAGA of INdAM and of CrypTO, the group of Cryptography and Number Theory of the Politecnico di Torino.

The work of Antonio J. Di Scala was partially supported by the QUBIP project (https://www.qubip.eu), funded by the European Union under the Horizon Europe framework programme [grant agreement no. 101119746].

This work was partially supported by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union – NextGenerationEU.

We would also like to thank Andrea Natale and Ricardo Pontaza for their insights and discussions, which helped us improve the analysis of our techniques. Finally, we thank the anonymous reviewers of a previous version of this manuscript who provided us with helpful comments and recommendations.

References

1. Alamati, N., De Feo, L., Montgomery, H., Patranabis, S.: Cryptographic group actions and applications. In: Moriai and Wang [26], pp. 411–439. https://doi.org/10.1007/978-3-030-64834-3_14
2. Baldi, M., Beckwith, A.B.L., Biasse, J.F., Esser, A., Gaj, K., Mohajerani, K., Pelosi, G., Persichetti, E., Saarinen, M.J.O., Santini, P., Wallace, R.: LESS (version 1.1). Tech. rep., National Institute of Standards and Technology (2023), https://www.less-project.com/
3. 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 (2019). https://doi.org/10.1109/ISIT.2019.8849855
4. Barenghi, A., Biasse, J., 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), https://doi.org/10.1080/23799927.2022.2048206
5. Barenghi, A., Biasse, J.F., Persichetti, E., Santini, P.: On the computational hardness of the code equivalence problem in cryptography. Advances in Mathematics of Communications **17**(1), 23–55 (2023), https://doi.org/10.3934/amc.2022064
6. Battagliola, M., Borin, G., Meneghetti, A., Persichetti, E.: Cutting the GRASS: Threshold GRoup Action Signature Schemes. Cryptology ePrint Archive, Paper 2023/859 (2023), https://eprint.iacr.org/2023/859
7. Battagliola, M., Borin, G., Meneghetti, A., Persichetti, E.: Cutting the grass: Threshold group action signature schemes. In: Oswald, E. (ed.) Topics in Cryptology – CT-RSA 2024. pp. 460–489. Springer Nature Switzerland, Cham (2024), https://doi.org/10.1007/978-3-031-58868-6_18
8. Benčina, B., Budroni, A., Chi-Domínguez, J.J., Kulkarni, M.: Properties of Lattice Isomorphism as a Cryptographic Group Action. In: International Conference on Post-Quantum Cryptography. pp. 170–201. Springer (2024), https://doi.org/10.1007/978-3-031-62743-9_6

9. Beullens, W.: Not enough LESS: An improved algorithm for solving code equivalence problems over $\mathbb{F}_q$. In: International Conference on Selected Areas in Cryptography. pp. 387–403. Springer (2020), https://doi.org/10.1007/978-3-030-81652-0_15
10. Beullens, W., Katsumata, S., Pintore, F.: Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In: Moriai and Wang [26], pp. 464–492. https://doi.org/10.1007/978-3-030-64834-3_16
11. Biasse, J.F., Micheli, G., Persichetti, E., Santini, P.: LESS is more: Code-based signatures without syndromes. In: Nitaj, A., Youssef, A.M. (eds.) AFRICACRYPT 20. LNCS, vol. 12174, pp. 45–65. Springer, Heidelberg (Jul 2020). https://doi.org/10.1007/978-3-030-51938-4_3
12. Bos, J.W., Bronchain, O., Ducas, L., Fehr, S., Huang, Y.H., Pornin, T., Postlethwaite, E.W., Prest, T., Pulles, L.N., van Woerden, W.: Hawk version 1.0 (june 1, 2023). Tech. rep., National Institute of Standards and Technology (2023), https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/hawk-spec-web.pdf
13. Budroni, A., Chi-Domínguez, J.J., D'Alconzo, G., Di Scala, A.J., Kulkarni, M.: `relaxed-lce-algorithms`, available at https://github.com/JJChiDguez/relaxed-lce-algorithms.git
14. Chavez-Saab, J., Santos, M.C.R., Feo, L.D., Eriksen, J.K., Hess, B., Kohel, D., Leroux, A., Longa, P., Meyer, M., Panny, L., Patranabis, S., Petit, C., Henríquez, F.R., Schaeffler, S., Wesolowski, B.: Sqisign version 1.0 (june 1, 2023). Tech. rep., National Institute of Standards and Technology (2023), https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/sqisign-spec-web.pdf
15. Chou, T., Niederhagen, R., Persichetti, E., Ran, L., Hajatiana, T., Reijnders, K., Samardjiska, S., Trimoska, M.: MEDS (version 1.1). Tech. rep., National Institute of Standards and Technology (2023), https://www.meds-pqc.org/
16. 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: Mrabet, N.E., Feo, L.D., Duquesne, S. (eds.) Progress in Cryptology - AFRICACRYPT 2023 - 14th International Conference on Cryptology in Africa, Sousse, Tunisia, July 19-21, 2023, Proceedings. Lecture Notes in Computer Science, vol. 14064, pp. 28–52. Springer (2023). https://doi.org/10.1007/978-3-031-37679-5_2
17. Chou, T., Persichetti, E., Santini, P.: On Linear Equivalence, Canonical Forms, and Digital Signatures. Cryptology ePrint Archive, Paper 2023/1533 (2023), https://eprint.iacr.org/2023/1533
18. Couveignes, J.M.: Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291 (2006), https://eprint.iacr.org/2006/291
19. D'Alconzo, G., Di Scala, A.J.: Representations of group actions and their applications in cryptography. Finite Fields and Their Applications **99**, 102476 (2024). https://doi.org/https://doi.org/10.1016/j.ffa.2024.102476
20. Gaborit, P., Otmani, A.: `TABLES OF SELF-DUAL CODES`, available at https://www.unilim.fr/pages_perso/philippe.gaborit/SD/
21. Gaborit, P., Otmani, A.: Experimental constructions of self-dual codes. Finite Fields and Their Applications **9**(3), 372–394 (2003). https://doi.org/https://doi.org/10.1016/S1071-5797(03)00011-X
22. Joux, A.: MPC in the head for isomorphisms and group actions. Cryptology ePrint Archive, Paper 2023/664 (2023), https://eprint.iacr.org/2023/664
23. Kazmi, R.A.: Cryptography from post-quantum assumptions. Cryptology ePrint Archive, Report 2015/376 (2015), https://eprint.iacr.org/2015/376

24. Leon, J.: Computing automorphism groups of error-correcting codes. IEEE Transactions on Information Theory **28**(3), 496–511 (1982), https://doi.org/10.1109/TIT.1982.1056498
25. Leroux, A., Roméas, M.: Updatable encryption from group actions. In: International Conference on Post-Quantum Cryptography. pp. 20–53. Springer (2024), https://doi.org/10.1007/978-3-031-62746-0_2
26. Moriai, S., Wang, H. (eds.): ASIACRYPT 2020, Part II, LNCS, vol. 12492. Springer, Heidelberg (Dec 2020)
27. National Institute of Standards and Technology: Post-Quantum Cryptography Standardization. https://csrc.nist.gov/projects/post-quantum-cryptography (2017)
28. National Institute of Standards and Technology: Post-quantum cryptography: Digital signature schemes. Round 1 Additional Signatures (2023), https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures
29. Persichetti, E., Randrianariso, T.H., Santini, P.: An attack on a non-interactive key exchange from code equivalence. Tatra Mountains Mathematical Publications **82**(2), 53–64 (2023), https://doi.org/10.2478/tmmp-2022-0018
30. Persichetti, E., Santini, P.: A New Formulation of the Linear Equivalence Problem and Shorter LESS Signatures. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023. pp. 351–378. Springer Nature Singapore, Singapore (2023), https://doi.org/10.1007/978-981-99-8739-9_12
31. Petrank, E., Roth, R.M.: Is code equivalence easy to decide? IEEE Transactions on Information Theory **43**(5), 1602–1604 (1997), https://doi.org/10.1109/18.623157
32. Reijnders, K., Samardjiska, S., Trimoska, M.: Hardness Estimates of the Code Equivalence Problem in the Rank Metric. Designs, Codes and Cryptography pp. 1–30 (01 2024). https://doi.org/10.1007/s10623-023-01338-x
33. Saeed, M.A.: Algebraic Approach for Code Equivalence. Ph.D. thesis, Normandie Université, University of Khartoum, (2017), Available at https://theses.hal.science/tel-01678829v2
34. Santini, P., Baldi, M., Chiaraluce, F.: Computational hardness of the permuted kernel and subcode equivalence problems. IEEE Transactions on Information Theory **70**(3), 2254–2270 (2024). https://doi.org/10.1109/TIT.2023.3323068
35. Sendrier, N.: On the dimension of the hull. SIAM Journal on Discrete Mathematics **10**(2), 282–293 (1997), https://doi.org/10.1137/S0895480195294027
36. Sendrier, N.: Finding the permutation between equivalent linear codes: the support splitting algorithm. IEEE Transactions on Information Theory **46**(4), 1193–1203 (2000). https://doi.org/10.1109/18.850662
37. Sendrier, N., Simos, D.E.: The hardness of code equivalence over $\mathbb{F}_q$ and its application to code-based cryptography. In: Gaborit, P. (ed.) Post-Quantum Cryptography - 5th International Workshop, PQCrypto 2013. pp. 203–216. Springer Heidelberg (June 2013), https://doi.org/10.1007/978-3-642-38616-9_14
38. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 9.8) (2023), https://www.sagemath.org