On Linear Equivalence, Canonical Forms, and Digital Signatures

Tung Chou[1], Edoardo Persichetti[2], and Paolo Santini[3]

[1] Academia Sinica, Taipei, Taiwan
[2] Florida Atlantic University, Boca Raton, USA and Sapienza University, Rome, Italy
[3] Marche Polytechnic University, Ancona, Italy

Abstract. The LESS signature scheme, introduced in 2020, represents a fresh research direction to obtain practical code-based signatures. LESS is based on the linear equivalence problem for codes, and the scheme is entirely described using matrices, which define both the codes, and the maps between them. It makes sense then, that the performance of the scheme depends on how efficiently such objects can be represented.
In this work, we investigate canonical forms for matrices, and how these can be used to obtain very compact signatures. We present a new notion of equivalence for codes, and prove that it reduces to linear equivalence; this means there is no security loss when applying canonical forms to LESS. Additionally, we flesh out a potential application of canonical forms to cryptanalysis, and conclude that this does not improve on existing attacks, for the regime of interest. Finally, we analyze the impact of our technique, showing that it yields a drastic reduction in signature size when compared to the LESS submission, resulting in the smallest sizes for code-based signature schemes based on zero-knowledge.

1 Introduction

LESS is a post-quantum signature scheme first introduced in [13]. The scheme is usually considered part of code-based cryptography, but it departs from the traditional methodology of this area. In fact, rather than exploiting the difficulty of decoding, LESS relies on the idea of finding some kind of isomorphism between linear codes. This notion is well-known in coding theory under the name of *code equivalence*. Interestingly, in the context of cryptography, such a concept of equivalence can be seen as a *group action*, akin to the ubiquitous one behind the Discrete Logarithm Problem (DLP), although showing more similarities to settings such as the isomorphisms between polynomials, or graphs. It is in this way that LESS is constructed, following in the steps of well-trodden paths to construct a Sigma protocol based on the Code Equivalence Problem; this is then turned into a signature scheme using the Fiat-Shamir transform.

Subsequent works followed, mainly trying to improve the efficiency of the protocol: for instance, the authors in [6] show that the signature size can be reduced by having a public key comprised of more than two equivalent codes, as well as by selecting challenges according to a fixed-weight distribution. These optimizations are generic, in the sense that they can be applied to any scheme following the same framework, and have appeared in literature in other works, such as for instance [17]. In [23], instead, the authors investigate optimizations that are specific to the chosen setting, i.e. that apply only to code equivalence. In the paper, it is shown that it is possible to further reduce the signature size, by a large factor, as some pieces of information in the commitments are redundant. The idea of [23] is later used for the specification of LESS [3], as submitted to NIST's call for additional post-quantum signatures [22]. The specification shows that the smallest signature sizes are around 5.0 KiB, 13.4 KiB, and 26.6 KiB for security categories 1, 3 and 5, respectively. These sizes are achieved by using more than 2 generator matrices in each public key, which increases the public key sizes by a large factor.

1.1 Our Contributions

In this paper, we introduce the concept of *canonical forms* for matrices and show how it can be applied to code equivalence. By canonical form, we refer to the representative of a certain equivalence class; the equivalence we focus on is the one derived from linear equivalence. We show that using canonical forms one can define a new notion of equivalence between codes, which we call Canonical Form Code Equivalence. We show that canonical forms turn out to be a rather useful tool, since they can greatly improve the performance of schemes such as LESS (namely, reducing the signature size) and, at the same time, could even be helpful in solving certain instances of code equivalence.

When the canonical form possesses some desirable properties (e.g., efficiently computable and low failure probability), one can show that, for random instances, the new formulation is as hard as the original one. This allows us to improve the LESS scheme, by using more efficient canonical forms for commitment and verification;

these effectively replace the rudimentary one used in the original works [6, 13] as well the more sophisticated one proposed in [23] (and incorporated as part of the LESS submission [3]). With this new variant, which we call CF-LESS, it is possible to achieve much smaller signatures: for instance, using the same code and protocol parameters as in "balanced" parameter sets from the LESS submission [3] (which uses only 2 generator matrices and aims to minimize the public key size) yields signature of only 2.4 KiB, 5.7 KiB, and 9.8 KiB for NIST security categories 1, 3 and 5, respectively.

As mentioned above, interestingly, we also found that the cost of exhaustive-search solvers for code equivalence can be reduced by a large factor by exploiting exactly the concept of canonical forms: for example, for the case of *linear equivalence*, the cost is intuitively reduced from $O\big(\sqrt{n!(q-1)^n}\big)$ to $O\big(\sqrt{\binom{n}{k}}\big)$ operations, where each operation has a cost which is polynomial in n, k and q. Nevertheless, the improvement does not affect the claimed security levels of the parameters of LESS.

1.2 Paper Organization

The paper is organized as follows. Section 2 specifies our notation and summarizes some preliminary notions. Section 3 describes the equivalence relations that we consider for matrices over $\mathbb{F}_q$, and introduces the concept of canonical forms. This serves as an important basis for the discussions in the sections that follow. Section 4 shows concrete ways to define canonical forms, starting from existing ones and including some entirely new formulations. In Section 5, we first briefly review the Sigma protocol underlying LESS, and then present a new Sigma protocol, which we refer to as the CF-LESS Sigma protocol, that makes use of canonical forms to reduce the communication size. We will see in Section 7 that this has a considerable impact on signature size, allowing for a drastic reduction which yields the smallest signature sizes among many post-quantum schemes, and in particular, code-based schemes based on zero-knowledge. Finally, in Section 6, we discuss the security of our new technique: we first argue that CF-LESS is secure by showing a security reduction, and then discuss an application of canonical forms to cryptanalysis, which results in an intuitive attack against LEP.

2 Notation and Preliminaries

As usual, $\mathbb{F}_q$ denotes the finite field with q elements and $\mathbb{F}_q^*$ stands for its multiplicative group. We use bold uppercase (resp., lowercase) letter for matrices (resp., vectors). Given a vector $\mathbf{v}$, we define $\mathsf{multiset}(\mathbf{v})$ as the multiset formed by the entries of $\mathbf{v}$. We use three main matrix groups in our work, which are all subgroups of the general linear group GL_n of non-singular matrices over $\mathbb{F}_q$. The first, is the symmetric group, i.e. the group of permutations on n objects, represented as binary matrices with a single 1 in each row and column; this is denoted by S_n. Second, we consider the group comprised of diagonal matrices over $\mathbb{F}_q$, such that all elements on the main diagonal are non-zero; we denote this by D_n. Finally, we define a group which is a generalization of S_n, where each permutation is scaled with non-zero factors from $\mathbb{F}_q$; we denote this by M_n. In other words, for each $\mathbf{Q} \in M_n$, we have that $\mathbf{Q} = \mathbf{P} \cdot \mathbf{D}$, where $\mathbf{P} \in S_n$ and $\mathbf{D} \in D_n$. Such matrices are known as *monomial matrices*, and we will thus refer to the group M_n as the *monomial group* over $\mathbb{F}_q$. In addition, we introduce a special type of permutations: given $k < n$, we call $S_{k,n} \subset S_n$ the set of permutations such that, for every $\mathbf{P} \in S_{k,n}$,

- $i < i' \leq k$ and $\mathbf{P}_{i,j} = \mathbf{P}_{i',j'} = 1$ implies that $j < j'$, and
- $k < i < i'$ and $\mathbf{P}_{i,j} = \mathbf{P}_{i',j'} = 1$ implies that $j < j'$.

In fact, once the first k columns of a matrix in $S_{k,n}$ is defined, the whole matrix is defined. Therefore, every matrix in $S_{k,n}$ can be represented as a vector in $\mathbb{F}_2^n$ of Hamming weight k, such that the row indices of the 1's in the first k columns are exactly the indices of the 1's in the vector.

The matrices considered in this work are mostly treated in the context of coding theory. A *linear code* $\mathscr{C}$ with *dimension* $k > 0$ and *length* $n > k$ is a k-dimensional linear subspace of $\mathbb{F}_q^n$ and, as such, can be represented compactly by a full rank matrix $\mathbf{G} \in \mathbb{F}_q^{k \times n}$ which serves as a basis, i.e., $\mathscr{C} = \left\{\mathbf{uG} : \mathbf{u} \in \mathbb{F}_q^k\right\}$. In the context of coding theory, this matrix is called *generator matrix*. Note that there are several choices for generator matrices for the same code, corresponding to different choices of basis: any $\mathbf{G}$ and $\mathbf{S} \cdot \mathbf{G}$, where $\mathbf{S} \in \mathrm{GL}_k$, generate the same code. Whenever a generator matrix is in the form $(\mathbf{I}_k \mid \mathbf{A})$, where $\mathbf{I}_k$ is the identity of size k, we say it is in *systematic*

form (or *standard form*). When $\{1, \cdots, k\}$ is an *information set* (i.e. corresponding columns are linearly independent), this form can be obtained by setting $\mathbf{S}$ as the inverse of the leftmost $k \times k$ submatrix in $\mathbf{G}$. Otherwise, let $\mathbf{G}_J$ be the submatrix of $\mathbf{G}$ with columns indexed by J. One can first compute the *Reduced Row-Echelon Form (RREF)*, i.e., the matrix $\mathbf{G}_J^{-1}\mathbf{G}$ with J being the first information set[4] (in lexicographical order), and then eventually applying a column permutation so that the identity columns are moved from positions J to $\{1, \cdots, k\}$. In other words, the systematic form is defined as

$$\mathsf{SF}(\mathbf{G}) = \left(\mathbf{I}_k \mid \mathbf{G}_J^{-1}\mathbf{G}_{\{1,\cdots,n\}\setminus J}\right), \;\; J \text{ is the first information set.}$$

Note that this also corresponds to $\mathsf{RREF}(\mathbf{G})\cdot\mathbf{P}$ for some permutation matrix $\mathbf{P} \in S_{k,n}$, where $\mathsf{RREF} : \mathbb{F}_q^{k\times n} \mapsto \mathbb{F}_q^{k\times n}$ is the function computing the RREF form. In the following, we will sometime use RREF^* to define the function returning both the systematic form, as well as the permutation $\mathbf{P} \in S_{k,n}$. SF and RREF are invariant under changes of basis: for any two generator matrices $\mathbf{G}$ and $\mathbf{S} \in \mathrm{GL}_k$, it holds that $\mathsf{SF}(\mathbf{G}) = \mathsf{SF}(\mathbf{SG})$ and $\mathsf{RREF}(\mathbf{G}) = \mathsf{RREF}(\mathbf{SG})$. If $\{1, \cdots, k\}$ is an information set, then SF and RREF coincide and $\mathbf{P} = \mathbf{I}_k$.

Linear codes are typically measured in the *Hamming metric*, which defines the *weight* of a (code)word as the number of its non-zero positions. Indeed, the precise distribution of weights of the various codewords, and the *distance* between them, is exactly what characterizes the code, enabling features such as error detection and correction. With this in mind, it is then natural to consider *equivalent* two codes having an identical weight distribution. Equivalent codes are obtained from one another via an *isometry*, i.e. a map preserving distances. In the simplest of cases, such a map consists of just a permutation, which leads to the notion of permutation equivalence; if instead the map is a monomial one, this is usually known as *linear equivalence*. The more general notion of *semilinear equivalence* refers to the addition of a field automorphism. This concept is not relevant for cryptographic applications, and we do not treat it here. It is immediate to note that permutation equivalence is nothing but a special case of linear equivalence. Equivalent codes can

[4] For instance, we consider the natural ordering defined by the relations $\{1, \cdots, k-1, k\} < \{1, \cdots, k-1, k+1\} < \cdots < \{1, \cdots, k-1, n\} < \cdots < \{1, \cdots, n-1, n\} < \cdots < \{2, \cdots, k, k+1\} < \cdots < \{n-k+1, \cdots, n\}$

be easily represented via their generator matrices, so that, if $\mathscr{C}$ is permutationally (resp. linearly) equivalent to $\mathscr{C}'$, then there exist a change-of-basis matrix $\mathbf{S} \in \mathrm{GL}_k$ and a permutation $\mathbf{P} \in S_n$ (resp. monomial $\mathbf{Q} \in M_n$) such that $\mathbf{G}' = \mathbf{S} \cdot \mathbf{G} \cdot \mathbf{P}$, where $\mathbf{G}$ and $\mathbf{G}'$ are the respective generator matrices.

Determining whether two codes are equivalent is often a challenging task. In cryptography, one is usually interested in the computational version of the problem, which we report below.

Problem 1 (Code Equivalence) *Given linear codes $\mathscr{C}, \mathscr{C}' \subseteq \mathbb{F}_q^n$ with dimension k, defined by generator matrices $\mathbf{G}, \mathbf{G}'$ respectively, decide if $\mathscr{C}$ is equivalent to $\mathscr{C}'$, i.e., if there exists $\mathbf{S} \in \mathrm{GL}_k(\mathbb{F}_q)$ and $\mathbf{P} \in S_n$ (resp. $\mathbf{Q} \in M_n$) such that $\mathbf{G}' = \mathbf{SGP}$ (resp. $\mathbf{G}' = \mathbf{SGQ}$).*

The problem is known respectively as *Permutation Equivalence Problem (PEP)* or *Linear Equivalence Problem (LEP)*, depending on which kind of equivalence is involved. The hardness of the code equivalence problem has been studied in detail through several works, and we point the interested reader to [7,8,25] for an extensive treatment.

3 Equivalence Relations on Matrices and Codes

Let $n, k, q \in \mathbb{Z}$ be the typical coding theory parameters, as defined above. The equivalence relations introduced in this sections are defined using a set $F \subseteq D_k \times S_k \times D_{n-k} \times S_{n-k}$. We consider 16 cases, where each component for F can be either the set of one identity matrix (of the respective size), or the entire group. For instance, F could be $\{\mathbf{I}_k\} \times \{\mathbf{I}_k\} \times \{\mathbf{I}_{n-k}\} \times S_{n-k}$, i.e. consisting only of column permutations, or $\{\mathbf{I}_k\} \times S_k \times \{\mathbf{I}_{n-k}\} \times S_{n-k}$, which considers permutations for both rows and columns, and so on. To each case, we associate a notion of equivalence relation on $\mathbb{F}_q^{k \times (n-k)}$ and $\mathbb{F}_q^{k \times n}$. As we shall see later, these notions can be used to boost the performance of schemes based on code equivalence and, at the same time, can be useful in the cryptanalysis for the problem.

3.1 New Equivalence Relations on Matrices

We first introduce a notion of equivalence on $\mathbb{F}_q^{k \times (n-k)}$.

Definition 1 (Equivalence Relations on $\mathbb{F}_q^{k\times(n-k)}$).
Given $F \subseteq D_k \times S_k \times D_{n-k} \times S_{n-k}$, we say that two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{F}_q^{k\times(n-k)}$ are equivalent with respect to R_F (denoted as $\mathbf{A} \sim_{\mathsf{R}_F} \mathbf{B}$) if

$$\mathbf{B} = \mathbf{P}_r \cdot \mathbf{D}_r \cdot \mathbf{A} \cdot \mathbf{D}_c \cdot \mathbf{P}_c,$$

for some $(\mathbf{D}_r, \mathbf{P}_r, \mathbf{D}_c, \mathbf{P}_c) \in F$ [5] *.*

Note that replacing $\mathbf{P}_r \cdot \mathbf{D}_r$ by $\mathbf{D}_r \cdot \mathbf{P}_r$ or replacing $\mathbf{D}_c \cdot \mathbf{P}_c$ by $\mathbf{P}_c \cdot \mathbf{D}_c$ does not change the definition of R_F. We now introduce a notion of equivalence on $\mathbb{F}_q^{k\times n}$.

Definition 2 (Equivalence Relations on $\mathbb{F}_q^{k\times n}$).
Given $F \subseteq D_k \times S_k \times D_{n-k} \times S_{n-k}$, we say that two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{F}_q^{k\times n}$ are equivalent with respect to R'_F (denoted as $\mathbf{A} \sim_{\mathsf{R}'_F} \mathbf{B}$) if

$$\mathbf{B} = \mathbf{T} \cdot \mathbf{A} \cdot \begin{bmatrix} \mathbf{D}_r^{-1}\mathbf{P}_r^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c \end{bmatrix},$$

where $(\mathbf{D}_r, \mathbf{P}_r, \mathbf{D}_c, \mathbf{P}_c) \in F$ and $\mathbf{T} \in \mathrm{GL}_k$.

Remark 1. $\mathsf{R}_{F'}$ is a subcase of linear equivalence since the monomial in the above definition has a very special structure: it never mixes the first k columns with the last $n-k$ ones.

Replacing $\mathbf{D}_r^{-1}\mathbf{P}_r^{-1}$ by $\mathbf{D}_r\mathbf{P}_r$ in the above definition does not change R'_F; we used $\mathbf{D}_r^{-1}\mathbf{P}_r^{-1}$ because this is convenient for explaining the relationship between R_F and R'_F. We do this in the next theorem.

Theorem 1. *Let $\mathbf{A}, \mathbf{B} \in \mathbb{F}_q^{k\times n}$ be two generator matrices for which $\{1, \cdots, k\}$ is an information set. Let $(\mathbf{I}_k \mid \widehat{\mathbf{A}})$ and $(\mathbf{I}_k \mid \widehat{\mathbf{B}})$ be the respective systematic generator matrices. Then*

$$\mathbf{A} \sim_{\mathsf{R}'_F} \mathbf{B} \iff \widehat{\mathbf{A}} \sim_{\mathsf{R}_F} \widehat{\mathbf{B}}.$$

Moreover,

$$\mathbf{B} = \mathbf{T} \cdot \mathbf{A} \cdot \begin{bmatrix} \mathbf{D}_r^{-1}\mathbf{P}_r^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c \end{bmatrix} \iff \widehat{\mathbf{B}} = \mathbf{P}_r \cdot \mathbf{D}_r \cdot \widehat{\mathbf{A}} \cdot \mathbf{D}_c \cdot \mathbf{P}_c,$$

where $(\mathbf{D}_r, \mathbf{P}_r, \mathbf{D}_c, \mathbf{P}_c) \in F$ and $\mathbf{T} \in \mathrm{GL}_k$.

[5] The subscripts r and c stand for row and column, respectively. We use these subscripts because in Definition 1, $\mathbf{P}_r, \mathbf{D}_r$ can be considered as row operations and $\mathbf{P}_c, \mathbf{D}_c$ can be considered as column operations.

Proof. Let $\mathbf{A} = (\mathbf{A}_1 \mid \mathbf{A}_2)$, with $\mathbf{A}_1 \in \mathrm{GL}_k$ and $\mathbf{A}_2 \in \mathbb{F}_q^{k\times(n-k)}$. Since by hypothesis $\{1, \cdots, k\}$ is an information set, the systematic form for $\mathbf{A}$ is $\mathbf{A}_1^{-1}\mathbf{A}$, so that $\widehat{\mathbf{A}} = \mathbf{A}_1^{-1} \cdot \mathbf{A}_2$. Using analogous notation for $\mathbf{B}$, we get $\widehat{\mathbf{B}} = \mathbf{B}_1^{-1} \cdot \mathbf{B}_2$. For the direction $\Longrightarrow$, it is enough to consider that $\mathbf{B}_1 = \mathbf{T} \cdot \mathbf{A}_1 \cdot \mathbf{D}_r^{-1} \cdot \mathbf{P}_r^{-1}$ and $\mathbf{B}_2 = \mathbf{T} \cdot \mathbf{A}_2 \cdot \mathbf{D}_c \cdot \mathbf{P}_c$, which implies

$$\begin{aligned}\widehat{\mathbf{B}} &= \underbrace{(\mathbf{T} \cdot \mathbf{A}_1 \cdot \mathbf{D}_r^{-1} \cdot \mathbf{P}_r^{-1})^{-1}}_{\mathbf{B}_1^{-1}} \cdot \underbrace{\mathbf{T} \cdot \mathbf{A}_2 \cdot \mathbf{D}_c \cdot \mathbf{P}_c}_{\mathbf{B}_2} \\ &= \mathbf{P}_r \cdot \mathbf{D}_r \cdot \underbrace{\mathbf{A}_1^{-1} \cdot \mathbf{A}_2}_{\widehat{\mathbf{A}}} \cdot \mathbf{D}_c \cdot \mathbf{P}_c.\end{aligned}$$

For the direction $\Longleftarrow$, we have

$$\begin{aligned}\mathbf{B}_1^{-1} \underbrace{\begin{bmatrix}\mathbf{B}_1 & \mathbf{B}_1 \cdot \widehat{\mathbf{B}}\end{bmatrix}}_{\mathbf{B}} &= \begin{bmatrix}\mathbf{I}_k & \widehat{\mathbf{B}}\end{bmatrix} = \begin{bmatrix}\mathbf{I}_k & \mathbf{P}_r \cdot \mathbf{D}_r \cdot \widehat{\mathbf{A}} \cdot \mathbf{D}_c \cdot \mathbf{P}_c\end{bmatrix} \\ &= \mathbf{P}_r \cdot \mathbf{D}_r \begin{bmatrix}\mathbf{I}_k & \widehat{\mathbf{A}}\end{bmatrix} \begin{bmatrix}\mathbf{D}_r^{-1} \cdot \mathbf{P}_r^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c \cdot \mathbf{P}_c\end{bmatrix},\end{aligned}$$

which implies

$$\underbrace{\begin{bmatrix}\mathbf{B}_1 & \mathbf{B}_1 \cdot \widehat{\mathbf{B}}\end{bmatrix}}_{\mathbf{B}} = \mathbf{B}_1 \cdot \mathbf{P}_r \cdot \mathbf{D}_r \cdot \mathbf{A}_1^{-1} \underbrace{\begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_1 \cdot \widehat{\mathbf{A}}\end{bmatrix}}_{\mathbf{A}} \begin{bmatrix}\mathbf{D}_r^{-1}\mathbf{P}_r^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c\end{bmatrix}.$$

□

3.2 Representatives for Equivalence Classes

In the previous section, we defined equivalence relations R_F and R'_F. For the purpose of this paper, we are interested in specifying representatives for some (not necessarily all [6]) equivalence classes. We call *representative* of an equivalence class the canonical form of any element in the equivalence class. If the representative is not specified for an equivalence class, then the canonical form does not exist for its elements. The mapping between any element and its canonical form can be formalized via a *canonical form* function.

[6] Usually people try to specify a representative for every single equivalence class. For some choices of F, it seems hard to specify a representative for every equivalence class of R_F and R'_F while achieving some nice properties we want, such as having an efficient canonical-form-deriving algorithm. Therefore, we relaxed the constraint.

Definition 3 (Canonical Form Function).
For an equivalence relation R *defined on a set* S*, we define a canonical form function as a function* $\mathsf{CF}_{\mathsf{R}} : S \mapsto \{\{\perp\} \cup S\}$ *satisfying the following properties:*

i) *for any* $\mathbf{A} \in S$, $\mathsf{CF}_{\mathsf{R}}(\mathbf{A}) \neq \perp \implies \mathbf{A} \sim_{\mathsf{R}} \mathsf{CF}_{\mathsf{R}}(\mathbf{A})$;
ii) *for any* $\mathbf{A}, \mathbf{B} \in S$, $\mathbf{A} \sim_{\mathsf{R}} \mathbf{B} \implies \mathsf{CF}_{\mathsf{R}}(\mathbf{A}) = \mathsf{CF}_{\mathsf{R}}(\mathbf{B})$.

We say CF_{R} *has success probability* γ *if* $\mathsf{CF}_{\mathsf{R}}(\mathbf{A}) \neq \perp$ *with probability* γ*, when* $\mathbf{A}$ *is drawn uniformly at random from* S*.*

Remark 2. The first property implies that $\mathsf{CF}_{\mathsf{R}}(\mathbf{A}) = \mathsf{CF}_{\mathsf{R}}(\mathbf{B}) = \perp \implies \mathbf{A} \sim_{\mathsf{R}} \mathbf{B}$. Notice that the bidirectional $\mathbf{A} \sim_{\mathsf{R}} \mathbf{B} \iff \mathsf{CF}(\mathbf{A}) = \mathsf{CF}(\mathbf{B})$ is hindered by the fact that the canonical form may not be defined for some matrices (i.e., the output of CF is $\perp$), and holds only if $\gamma = 1$, i.e., the canonical form is defined for all elements in S.

As we shall see next, the above properties are crucial to have a tight correspondence between the classical code equivalence problem and the problems we rely on in this paper.

Proposition 1. *Assume that* $\mathsf{CF}_{\mathsf{R}_F}$ *is a canonical form function with* $\mathsf{R} = \mathsf{R}_F$ *and* $S = \mathbb{F}_q^{k \times (n-k)}$*. For any* $\mathbf{A} = (\mathbf{A}_1 \mid \mathbf{A}_2) \in \mathbb{F}_q^{k \times n}$ *where* $\mathbf{A}_1 \in \mathrm{GL}_k$*, we define* $\mathsf{CF}_{\mathsf{R}'_F}(\mathbf{A})$ *as* $\big(\mathbf{I}_k \mid \mathsf{CF}_{\mathsf{R}_F}(\mathbf{A}_1^{-1}\mathbf{A}_2)\big)$ *if* $\mathsf{CF}_{\mathsf{R}_F}(\mathbf{A}_1^{-1}\mathbf{A}_2) \neq \perp$*, or* $\perp$ *otherwise. Then* $\mathsf{CF}_{\mathsf{R}'_F}$ *is a canonical form function with* $\mathsf{R} = \mathsf{R}'_F$ *and* $S = \big\{(\mathbf{A}_1 \mid \mathbf{A}_2) \in \mathbb{F}_q^{k \times n} \mid \mathbf{A}_1 \in \mathrm{GL}_k\big\}$.

Proof. According to Theorem 1, $\mathsf{CF}_{\mathsf{R}'_F}(\mathbf{A}) \neq \perp$ implies

$$\mathsf{CF}_{\mathsf{R}'_F}(\mathbf{A}) = \big(\mathbf{I}_k \mid \mathsf{CF}_{\mathsf{R}_F}(\mathbf{A}_1^{-1}\mathbf{A}_2)\big) \sim_{\mathsf{R}'_F} (\mathbf{A}_1 \mid \mathbf{A}_2)$$

and

$$\begin{aligned}(\mathbf{A}_1 \mid \mathbf{A}_2) \sim_{\mathsf{R}'_F} (\mathbf{B}_1 \mid \mathbf{B}_2) &\implies \mathbf{A}_1^{-1}\mathbf{A}_2 \sim_{\mathsf{R}_F} \mathbf{B}_1^{-1}\mathbf{B}_2 \\ &\implies \mathsf{CF}_{\mathsf{R}_F}(\mathbf{A}_1^{-1}\mathbf{A}_2) = \mathsf{CF}_{\mathsf{R}_F}(\mathbf{B}_1^{-1}\mathbf{B}_2) \\ &\implies \mathsf{CF}_{\mathsf{R}'_F}((\mathbf{A}_1 \mid \mathbf{A}_2)) = \mathsf{CF}_{\mathsf{R}'_F}((\mathbf{B}_1 \mid \mathbf{B}_2)),\end{aligned}$$

so the properties in Definition 3 are satisfied. □

In the next section we show practical examples for $\mathsf{CF}_{\mathsf{R}_F}$. Note that each example of $\mathsf{CF}_{\mathsf{R}_F}$ immediately gives us a canonical form function $\mathsf{CF}_{\mathsf{R}'_F}$, as defined in the proposition above. In the remainder of the paper, we will often omit the subscripts R_F and R'_F, as it is always clear from the context which equivalence relation we are considering.

4 Defining Canonical Forms for R_F

This section shows how one can define canonical forms for R_F for different choices of F. Note that the first two approaches have been implicitly used in [23] and [3], while the remaining ones are entirely new. The number of field operations taken by each algorithm presented in this section is polynomial in q and n in the worst case.

Case 1. Let $F = \{\mathbf{I}_k\} \times \{\mathbf{I}_k\} \times \{\mathbf{I}_{n-k}\} \times S_{n-k}$. In this case, one can define the canonical form of any equivalence class as the matrix in which the columns are sorted with respect to a total ordering defined on $\mathbb{F}_q^k$. For example, the total ordering can simply be the lexicographic order. This way, the canonical form is well-defined for every equivalence class, and from any matrix the canonical form can be derived easily by simply sorting the columns.

Case 2. Let $F = \{\mathbf{I}_k\} \times \{\mathbf{I}_k\} \times D_{n-k} \times S_{n-k}$. Here, one can define the canonical form as the unique matrix such that 1) the first non-zero entry of each non-zero column is 1 and 2) the columns are sorted with respect to a total ordering defined on $\mathbb{F}_q^k$. Again, this guarantees that the canonical form is well-defined for every equivalence class. Given any matrix, to compute the canonical form, one can first scale each non-zero column to turn the first non-zero entry into 1, and then sort the columns.

Case 3. Let $F = \{\mathbf{I}_k\} \times S_k \times \{\mathbf{I}_{n-k}\} \times S_{n-k}$. Given a total ordering defined all size-$(n-k)$ multisets of elements in $\mathbb{F}_q$ and a total ordering defined on $\mathbb{F}_q^k$, for any equivalence class, we define the canonical form as the matrix of which the multisets for the rows are sorted and the columns are sorted, if it exists. For some equivalence classes, such a matrix might not exist. This happens only when there are two distinct rows that cannot be compared (because they lead to the same multiset) in all matrices in the equivalence class.

For any matrix, one can derive the corresponding canonical form by sorting first the rows using the partial ordering, and then the columns using the total ordering. Similarly, one can also define the

canonical form as the result of sorting first the columns using a partial ordering defined on $\mathbb{F}_q^k$, and then the rows using a total ordering defined on $\mathbb{F}_q^{n-k}$. It is easy to detect whether the corresponding canonical form exists or not, during the step of sorting rows.

One reasonably efficient way to define the total ordering for multisets is as follows: for any distinct multisets S and S', let $\mathbf{u}, \mathbf{u}'$ be the vectors obtained by sorting the elements in S and S', respectively. Then $S \leq S' \iff \mathbf{u} \leq \mathbf{u}'$, where $\mathbf{u}$ and $\mathbf{u}'$ are compared using a total ordering defined on $\mathbb{F}_q^{n-k}$. The algorithm above takes about $O(n^2)$ field operations, if logarithmic terms are omitted. A proven lower bound on the success probability of the algorithm above is shown in Appendix B.

Case 4. Let $F = \{\mathbf{I}_k\} \times S_k \times D_{n-k} \times S_{n-k}$ and choose d distinct integers $m_1, \ldots, m_d \in \mathbb{Z}$ such that

- $0 < m_1 < m_2 < \cdots < m_d < q-1$,
- $\gcd(m_i, q-1) = 1$ for all i, and
- $\mathrm{char}(\mathbb{F}_q)$ is not a factor of any m_i.

For efficiency of the algorithm that will be introduced below, it is reasonable to select m_i's as the smallest d integers that satisfy the criteria. We define the canonical form of a equivalence class as the matrix (if it exists) such that

1. for each non-zero column $\mathbf{v}$, the vector $(\sum_i \mathbf{v}_i^{m_1}, \ldots, \sum_i \mathbf{v}_i^{m_d})$ is a non-zero vector where the first non-zero entry is 1, and
2. the rows and columns are sorted in the way described for Case 3,

To derive the systematic form of a matrix, one can carry out one step to ensure that the first constraint above holds and then another step to ensure that the second constraint holds. The second step can be carried out by the algorithm in Case 3. The first step can be carried out as follows. For each non-zero column $\mathbf{v} \in \mathbb{F}_q^k$, we find the smallest j such that $\sum_i \mathbf{v}_i^{m_j} \neq 0$. If no such j exists, $\perp$ is returned. Otherwise, $\mathbf{v}$ is replaced by $(\sum_i \mathbf{v}_i^{m_j})^{\frac{-1}{m_j}} \cdot \mathbf{v}$. Note that $f(x) = x^{\frac{1}{m_j}}$ is a bijective [7] map on $\mathbb{F}_q^*$ and can be evaluated quickly [8] with a precomputed table of $q-1$ field elements.

[7] It is bijective because $\gcd(m_i, q-1) = 1$.
[8] Here we assume that each table lookup takes constant time.

The algorithm above essentially takes $O(d \cdot n^2)$ field operations with precomputed tables, if logarithmic terms are omitted. The algorithm is expected to take $O(n^2)$ field operations on average, as j is highly biased towards 1. Section 7 shows some experiment results regarding success probability of the algorithm.

Case 5. Let $F = D_k \times S_k \times D_{n-k} \times S_{n-k}$. The way we define canonical forms in this case is a little less intuitive, so we limit ourselves to describe here just the algorithm for deriving the canonical form from any matrix. Consider a matrix $\mathbf{A}$, for each j such that column j consists of only non-zero elements; we define $\mathbf{A}^{(j)}$ as the result of scaling the rows of $\mathbf{A}$ so that column j of $\mathbf{A}^{(j)}$ is $(1, 1, \ldots, 1)$. If every column contains 0, return $\perp$. Then, for each $\mathbf{A}^{(j)}$, the algorithm for Case 4 is applied to obtain a matrix $\mathbf{A}^{[j]}$ (if $\perp$ is not returned). Finally, the canonical form is simply defined as the smallest $\mathbf{A}^{[j]}$ with respect to a total ordering defined on $\mathbb{F}_q^{k \times (n-k)}$.

It is easy to see that the above procedure leads to a canonical form satisfying the first property in Definition 3. To show that the second property is also satisfied, we introduce two results.

Proposition 2. *Given $\mathbf{A}, \mathbf{B} \in \mathbb{F}_q^{k \times (n-k)}$ satisfying $\mathbf{B} = \mathbf{D}_r \cdot \mathbf{A} \cdot \mathbf{D}_c$ for some $\mathbf{D}_r, \mathbf{D}_c \in D_{n-k}$, such that column j of $\mathbf{A}$ (and $\mathbf{B}$) consists of only non-zero elements. Then $\mathbf{B}^{(j)} = \mathbf{A}^{(j)} \cdot \mathbf{D}'_c$ for some $\mathbf{D}'_c \in D_{n-k}$.*

Proof. Let the elements on the main diagonal of $\mathbf{D}_r$ be $x_1, \ldots x_k$. Let the elements on the main diagonal of $\mathbf{D}_c$ be $y_1, \ldots y_{n-k}$. Then column i of $\mathbf{A}^{(j)}$ is $(\mathbf{A}_{1,i} \cdot \mathbf{A}_{1,j}^{-1}, \ldots, \mathbf{A}_{k,i} \cdot \mathbf{A}_{k,j}^{-1})$, while column i of $\mathbf{B}^{(j)}$ is

$$(\frac{x_1 \cdot \mathbf{A}_{1,i} \cdot y_i}{x_1 \cdot \mathbf{A}_{1,j} \cdot y_j}, \ldots, \frac{x_k \cdot \mathbf{A}_{k,i} \cdot y_i}{x_k \cdot \mathbf{A}_{k,j} \cdot y_j}).$$

□

Proposition 3. *Given $\mathbf{A}, \mathbf{B} \in \mathbb{F}_q^{k \times (n-k)}$ satisfying $\mathbf{A} = \mathbf{P}_r^{-1} \cdot \mathbf{B} \cdot \mathbf{P}_c^{-1}$ for some $\mathbf{P}_r, \mathbf{P}_c \in S_{n-k}$, such that column j of $\mathbf{A}$ consists of only non-zero elements. Let i' be the row index of 1 in column i of $\mathbf{P}_c^{-1}$. In other words, the column permutation represented by $\mathbf{P}_c^{-1}$ maps column i' to column i. Then $\mathbf{A}^{(i)} = \mathbf{P}_r^{-1} \cdot \mathbf{B}^{(i')} \cdot \mathbf{P}_c^{-1}$.*

The proof for Proposition 3 is immediate and therefore omitted in the interest of space. Now, combining the two propositions, we have

$$\begin{aligned}\mathbf{B} = \mathbf{P}_r \cdot \mathbf{D}_r \cdot \mathbf{A} \cdot \mathbf{D}_c \cdot \mathbf{P}_c &\implies \mathbf{P}_r^{-1} \cdot \mathbf{B} \cdot \mathbf{P}_c^{-1} = \mathbf{D}_r \cdot \mathbf{A} \cdot \mathbf{D}_c \\ &\implies \mathbf{P}_r^{-1} \cdot \mathbf{B}^{(i')} \cdot \mathbf{P}_c^{-1} = \mathbf{A}^{(i)} \cdot \mathbf{D}'_c \\ &\implies \mathbf{B}^{(i')} = \mathbf{P}_r \cdot \mathbf{A}^{(i)} \cdot \mathbf{D}'_c \cdot \mathbf{P}_c.\end{aligned}$$

After applying the algorithm in Case 4 to $\mathbf{A}^{(i)}$ and $\mathbf{B}^{(i')}$, we obtain $\mathbf{A}^{[i]} = \mathbf{B}^{[i']}$. Therefore, the set of $\mathbf{A}^{[j]}$'s is the same as the set of $\mathbf{B}^{[j]}$'s, and we conclude that the algorithm leads to the same output for any $\mathbf{A}, \mathbf{B}$ in the same equivalence class.

The algorithm above takes about $O(d \cdot n^3)$ field operations, if logarithmic factors are omitted. The average-case running time is expected to be much better than the worst-case running time. Indeed, this is true for the algorithm in Case 4. Also, compared to the wost-case complexity $O(d \cdot n^2)$ for the previous case, the extra factor n comes from the number of columns consisting of only non-zero entries, which is at most $n - k$ and is, for "real" parameters, much smaller than $n - k$ on average. Section 7 shows some experiment results regarding success probability of the algorithm.

5 Application to LESS Signatures

In this section, we discuss the application of our technique to LESS.

5.1 The LESS Sigma Protocol

We begin by recalling, in Figure 1, the Sigma protocol underlying the LESS signature scheme.

As shown in [13], the protocol is *2-special sound*, with soundness error $\varepsilon = \frac{1}{2}$. Note that, if the matrices $\mathbf{Q}$ and $\widetilde{\mathbf{Q}}$ are both permutations, this protocol falls into a special case, in which security relies exclusively on PEP (as this is a special case of LEP); this may require some slight changes in how the protocol is actually deployed (for example, utilizing different parameters or particular choices of codes, such as self-orthogonal codes).

Private Key: Matrix $\mathbf{Q} \in M_n$
Public Key: Generator Matrices $\mathbf{G}, \mathbf{G}' = \mathsf{RREF}(\mathbf{GQ}) \in \mathbb{F}_q^{k \times n}$

PROVER		VERIFIER
$\widetilde{\mathbf{Q}} \xleftarrow{\$} M_n$		
$\widetilde{\mathbf{G}} \leftarrow \mathbf{G} \cdot \widetilde{\mathbf{Q}}$		
$\mathtt{cmt} \leftarrow \mathsf{Hash}\big(\mathsf{RREF}(\widetilde{\mathbf{G}})\big)$	$\xrightarrow{\mathtt{cmt}}$	
	$\xleftarrow{\mathtt{ch}}$	$\mathtt{ch} \xleftarrow{\$} \{0,1\}$
If $\mathtt{ch} = 0$:		
$\mathtt{rsp} \leftarrow \widetilde{\mathbf{Q}}$		
Else:		
$\mathtt{rsp} \leftarrow \mathbf{Q}' \coloneqq \mathbf{Q}^{-1}\widetilde{\mathbf{Q}}$	$\xrightarrow{\mathtt{rsp}}$	**If** $\mathtt{ch} = 0$:
		Verify $\mathsf{Hash}\big(\mathsf{RREF}(\mathbf{G} \cdot \mathtt{rsp})\big) = \mathtt{cmt}$
		Else:
		Verify $\mathsf{Hash}\big(\mathsf{RREF}(\mathbf{G}' \cdot \mathtt{rsp})\big) = \mathtt{cmt}$

Fig. 1. The original LESS Sigma protocol

When $\mathtt{ch} = 0$, the verifier computes $\mathbf{G} \cdot \widetilde{\mathbf{Q}}$, which is the very same matrix $\widetilde{\mathbf{G}}$ computed by the prover. However, when $\mathtt{ch} = 1$, the verifier computes $\mathbf{G}' \cdot \mathbf{Q}'$, which is equal to $\widetilde{\mathbf{G}}$ up to a change of a basis. To avoid this discrepancy, and enable verification, the protocol computes the RREF, which is used as an invariant under change of basis.Note that, in the case $\mathtt{ch} = 0$, the response consists of the randomly-generated matrix $\widetilde{\mathbf{Q}}$, and can thus be compressed by transmitting only a seed for a secure PRNG, as is common practice.

Remark 3. The scheme presented in Figure 1 is simply the "core" element in the design of the LESS signature scheme. Indeed, to obtain a signature scheme, it is necessary to iterate the protocol, say t times, and apply the Fiat-Shamir transformation. Furthermore, a variety of optimizations are incorporated into the design, to improve the overall performance. For instance, the final signature scheme uses a variable number s of public keys (generating a tradeoff between public key and signature size); an " unbalanced" challenge string of fixed Hamming weight w (to maximize the reduction obtained by transmitting seeds for random objects) and a *seed tree* to compactly transmit seeds (as described in various previous works such as [11,12]). It is worth clarifying that the Fiat-Shamir transformation directly yields EUF-CMA security [18], and the addition of such standard optimizations does not affect this claim, as shown for instance in [6,17].

Next, we show how the use of canonical forms can be embedded into the protocol. The high-level intuition is that using canonical forms, on top of the RREF computation, enriches the invariance properties we are able to achieve. Technically, we let the prover and the verifier end up in two generator matrices which are equal up to an equivalence $\mathsf{R}_{F'}$; in other words, we allow the verifier to compute a code which is equivalent to the one generated by $\widetilde{\mathbf{G}}$, but not exactly the same. Leveraging this fact, the prover can provide multiple responses to verify the same commitment: among all such choices, we consider the one having the smallest communication cost.

5.2 The CF-LESS Sigma protocol

The Sigma protocol for CF-LESS is shown in Figure 2.

Private Key: Monomial $\mathbf{Q} \in M_n$
Public Key: Generator Matrices $\mathbf{G}, \mathbf{G}' = \mathsf{RREF}(\mathbf{GQ}) \in \mathbb{F}_q^{k \times n}$

PROVER		VERIFIER
do:		
$\widetilde{\mathbf{Q}} \xleftarrow{\$} M_n$		
$\widetilde{\mathbf{G}}, \mathbf{P} \leftarrow \mathsf{RREF}^*(\mathbf{G} \cdot \widetilde{\mathbf{Q}})$		
while $\mathsf{CF}(\widetilde{\mathbf{G}}) = \perp$		
$\mathtt{cmt} \leftarrow \mathsf{Hash}(\mathsf{CF}(\widetilde{\mathbf{G}}))$	$\xrightarrow{\mathtt{cmt}}$	
	$\xleftarrow{\mathtt{ch}}$	$\mathtt{ch} \xleftarrow{\$} \{0,1\}$
$\overline{\mathbf{Q}} \leftarrow \widetilde{\mathbf{Q}} \cdot \mathbf{P}$		
If $\mathtt{ch} = 0$:		
$\mathtt{rsp} \leftarrow \widetilde{\mathbf{Q}}$		
Else:		
$\mathtt{rsp} \leftarrow \mathsf{Compress}(\mathbf{Q}' := \mathbf{Q}^{-1}\overline{\mathbf{Q}})$	$\xrightarrow{\mathtt{rsp}}$	**If** $\mathtt{ch} = 0$:
		$\widetilde{\mathbf{G}}, \mathbf{P} \leftarrow \mathsf{RREF}^*(\mathbf{G} \cdot \mathtt{rsp})$
		Verify $\mathsf{Hash}(\mathsf{CF}(\widetilde{\mathbf{G}}))=\mathtt{cmt}$
		Else:
		Verify $\mathsf{Hash}(\mathsf{CF}(\mathbf{G}' \cdot \mathtt{rsp}))=\mathtt{cmt}$

Fig. 2. The CF-LESS Sigma protocol.

It is easy to see that the protocol is apparently complete if Compress is omitted. The function Compress is used to reduce the size of the response when $\mathtt{ch} = 1$ (and thus the signature size). Before specifying how Compress works, we show a way to decompose each matrix in S_n and M_n into several matrices.

Theorem 2. *For every* $\mathbf{P} \in S_n$*, we have* $\mathbf{P} = \mathbf{P}_{\text{is}} \cdot \begin{bmatrix} \mathbf{P}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_c \end{bmatrix}$*, for some* $(\mathbf{P}_{\text{is}}, \mathbf{P}_r, \mathbf{P}_c) \in S_{k,n} \times S_k \times S_{n-k}$ [9] *. For every* $\mathbf{M} \in M_n$*, we have* $\mathbf{M} = \mathbf{P}_{\text{is}} \cdot \begin{bmatrix} \mathbf{D}_r\mathbf{P}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c \end{bmatrix}$*, for some* $(\mathbf{P}_{\text{is}}, \mathbf{P}_r, \mathbf{P}_c, \mathbf{D}_r, \mathbf{D}_c) \in S_{k,n} \times S_k \times S_{n-k} \times D_k \times D_{n-k}$.

The way the function Compress operates is formalized next.

Definition 4. *Let* $\overline{F} \subset D_k \times S_k \times D_{n-k} \times S_{n-k}$ *such that each of its components is trivial if the corresponding component in* F *is non trivial, and vice versa. The function Compress:* $M_n \mapsto M_n$ *decomposes the input matrix into the form shown in Theorem 2 and replaces* $\mathbf{D}_r, \mathbf{P}_r, \mathbf{D}_c$*, or* $\mathbf{P}_c$ *by an identity matrix if the corresponding component of* $\overline{F}$ *is trivial.*

Let us make an example by considering $F = \{\mathbf{I}_k\} \times S_k \times D_{n-k} \times S_{n-k}$ for example. In this case, $\overline{F} = D_k \times \{\mathbf{I}_k\} \times \{\mathbf{I}_{n-k}\} \times \{\mathbf{I}_{n-k}\}$ and Compress replaces $\mathbf{P}_r$ by $\mathbf{I}_k$ and $\mathbf{D}_c, \mathbf{P}_c$ by $\mathbf{I}_{n-k}$. Therefore, we have

$$\mathsf{Compress}(\mathbf{Q}') = \mathsf{Compress}\left(\mathbf{P}_{\text{is}} \cdot \begin{bmatrix} \mathbf{D}_r\mathbf{P}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c \end{bmatrix}\right) = \mathbf{P}_{\text{is}} \cdot \begin{bmatrix} \mathbf{D}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n-k} \end{bmatrix}.$$

Without Compress, the response size for $\mathtt{ch} = 1$ would be $n(\lceil \log_2(n) \rceil + \lceil \log_2(q-1) \rceil)$ bits. Instead, as it turns out, $\mathsf{Compress}(\mathbf{Q}')$ takes fewer bits to represent than $\mathbf{Q}'$. We summarize the sizes we obtain in Table 1, for different choices of F. In Section 7, we use the formulas in Table 1 to derive the corresponding signature sizes of CF-LESS.

To summarize, below we report the number of bits required to represent $\mathsf{Compress}(\mathbf{Q}')$, for various choices of the set F.

Type	F	**Case**	isobits
Mono	$\{\mathbf{I}_k\} \times \{\mathbf{I}_k\} \times D_{n-k} \times S_{n-k}$	2	$k \cdot (\lceil \log_2(n) \rceil + \lceil \log_2(q-1) \rceil)$
Mono	$\{\mathbf{I}_k\} \times S_k \times D_{n-k} \times S_{n-k}$	4	$(n + k \cdot \lceil \log_2(q-1) \rceil)$
Mono	$D_k \times S_k \times D_{n-k} \times S_{n-k}$	5	n
Perm	$\{\mathbf{I}_k\} \times S_k \times \{\mathbf{I}_{n-k}\} \times S_{n-k}$	3	n

Table 1. isobits, the number of bits required to represent $\mathsf{Compress}(\mathbf{Q}')$.

[9] The subscript "is" stands for information set.

5.3 Properties of the CF-LESS Sigma Protocol

We are now ready to show that the CF-LESS Sigma protocol achieves the three fundamental properties for a ZK proof of knowledge, that is, completeness, zero-knowledge and special soundness. The first two properties are immediate; nonetheless, we prove them for the sake of completeness. For what concerns special soundness, we show that it reduces to finding solutions to the following problem

Problem 2 (Canonical Forms - Code Equivalence) *Let* $F \subseteq D_k \times S_k \times D_{n-k} \times S_{n-k}$. *Given* $\mathbf{G}, \mathbf{G}' \in \mathbb{F}_q^{k \times n}$, *find* $\mathbf{P}_{\text{is}}, \mathbf{P}'_{\text{is}} \in S_{k,n}$ *and* $(\mathbf{D}_r, \mathbf{P}_r, \mathbf{D}_c, \mathbf{P}_c), (\mathbf{D}'_r, \mathbf{P}'_r, \mathbf{D}'_c, \mathbf{P}'_c) \in \overline{F}$ *such that*

$$\mathsf{CF}\left(\mathbf{G} \cdot \mathbf{P}_{\text{is}} \cdot \begin{bmatrix} \mathbf{D}_r\mathbf{P}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c \end{bmatrix}\right) = \mathsf{CF}\left(\mathbf{G}' \cdot \mathbf{P}'_{\text{is}} \cdot \begin{bmatrix} \mathbf{D}'_r\mathbf{P}'_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}'_c\mathbf{P}'_c \end{bmatrix}\right).$$

For analogy with the traditional case, we refer to the above problem as *CF-LEP*; we will carefully analyze its hardness in Section 6.

Completeness and Zero-Knowledge. Zero-knowledge follows immediately from the fact that $\widetilde{\mathbf{Q}}$ is uniformly distributed over M_n.We now show why the protocol is also complete. To this end, we consider that when $\mathtt{ch} = 0$, the verifier receives $\widetilde{\mathbf{Q}}$ and executes the very same operations which have been performed by the prover. When $\mathtt{ch} = 1$, let us take $F = \{\mathbf{I}_k\} \times S_k \times D_{n-k} \times S_{n-k}$ for example. In this case, the verifier receives $\mathsf{Compress}(\mathbf{Q}')$, which is built such that

$$\mathbf{G}'\mathsf{Compress}(\mathbf{Q}') = \mathbf{G}'\mathbf{P}_{\text{is}} \begin{bmatrix} \mathbf{D}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n-k} \end{bmatrix} \sim_{\mathsf{R}'_F} \mathbf{G}'\mathbf{P}_{\text{is}} \begin{bmatrix} \mathbf{D}_r\mathbf{P}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c \end{bmatrix} = \mathbf{G}'\mathbf{Q}'$$
$$\implies \mathsf{CF}(\mathbf{G}' \cdot \mathsf{Compress}(\mathbf{Q}')) = \mathsf{CF}(\mathbf{G}' \cdot \mathbf{Q}').$$

Therefore, the CF-LESS protocol is complete for the specific F. Similarly, the protocol can be shown to be complete for $F = \{\mathbf{I}_k\} \times \{\mathbf{I}_k\} \times D_{n-k} \times S_{n-k}$ [10] and $F = D_k \times S_k \times D_{n-k} \times S_{n-k}$. It is also easy to see that the protocol is zero-knowledge and complete for the special case of permutations, when $F = \{\mathbf{I}_k\} \times S_k \times \{\mathbf{I}_{n-k}\} \times S_{n-k}$.

[10] The papers [3,6] make use of the same technique for $F = \{\mathbf{I}_k\} \times \{\mathbf{I}_k\} \times D_{n-k} \times S_{n-k}$.

Special Soundness. We show that the protocol is 2-special sound; given that it is a Sigma protocol with challenge space 1/2, it follows that it has soundness error $\varepsilon = 1/2$.

Proposition 4. *The protocol in Figure 2 is 2-special sound.*

Proof. We consider two accepting transcripts $\widetilde{\mathbf{Q}}$ and $\mathbf{Q}'$, respectively for $\mathtt{ch} = 0$ and $\mathtt{ch} = 1$, and commitment $\mathtt{cmt}$. From the knowledge of $\widetilde{\mathbf{Q}}$, one can obtain $\mathbf{P}$ and compute $\overline{\mathbf{Q}} = \widetilde{\mathbf{Q}} \cdot \mathbf{P}$. This implies that

$$\mathsf{Hash}\left(\mathsf{CF}(\mathbf{G} \cdot \overline{\mathbf{Q}})\right) = \mathsf{Hash}\big(\mathsf{CF}(\mathbf{G}' \cdot \mathbf{Q}')\big).$$

Either $\mathsf{CF}(\mathbf{G} \cdot \overline{\mathbf{Q}}) \neq \mathsf{CF}(\mathbf{G}' \cdot \mathbf{Q}')$ and a hash collision has been found, or $\mathsf{CF}(\mathbf{G} \cdot \overline{\mathbf{Q}}) = \mathsf{CF}(\mathbf{G}' \cdot \mathbf{Q}')$. Notice that $\mathbf{Q}'$ is the output of $\mathsf{Compress}$, so it is constituted by $\mathbf{P}'_{\text{is}} \in S_{k,n}$ and $(\mathbf{D}'_r, \mathbf{P}'_r, \mathbf{D}'_c, \mathbf{P}'_c) \in \overline{F}$. One can compute $\mathsf{Compress}$, on input $\overline{\mathbf{Q}}$, and find $\mathbf{P}_{\text{is}} \in S_{k,n}$ and $(\mathbf{D}_r, \mathbf{P}_r, \mathbf{D}_c, \mathbf{P}_c) \in \overline{F}$ such that

$$\mathsf{CF}(\mathbf{G} \cdot \overline{\mathbf{Q}}) = \mathsf{CF}\left(\mathbf{G} \cdot \mathbf{P}_{\text{is}} \cdot \begin{bmatrix} \mathbf{D}_r\mathbf{P}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c \end{bmatrix}\right).$$

So, we have obtained

$$\mathsf{CF}\left(\mathbf{G} \cdot \mathbf{P}_{\text{is}} \cdot \begin{bmatrix} \mathbf{D}_r\mathbf{P}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c\mathbf{P}_c \end{bmatrix}\right) = \mathsf{CF}\left(\mathbf{G}' \cdot \mathbf{P}'_{\text{is}} \cdot \begin{bmatrix} \mathbf{D}'_r\mathbf{P}'_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}'_c\mathbf{P}'_c \end{bmatrix}\right),$$

which implies that we have found a solution for CF-code equivalence, on input $\mathbf{G} \cdot \overline{\mathbf{Q}}$ and $\mathbf{G}'$. Starting from this solution, one can easily recover the equivalence between $\mathbf{G}$ and $\mathbf{G}'$. □

5.4 Computational Complexity

We briefly comment on the computational cost of the protocol in Figure 2. To this end, we rely on a heuristic which is commonly employed when studying code-based problems (e.g., in papers about information-set decoding). Most importantly, numerical simulations confirm the heuristic.

Heuristic 1 *Let $\mathbf{G} \in \mathbb{F}_q^{k \times n}$ be the generator matrix for a code with dimension k and length n. For any set $J \subseteq \{1, \cdots, n\}$ of size k, we consider that $\mathbf{G}_J$ is a $k \times k$ matrix sampled according to the uniform distribution over $\mathbb{F}_q$. Analogously, also $\mathbf{G}_{\{1,\cdots,n\}\setminus J}$ is a $k \times (n-k)$ matrix sampled according to the uniform distribution over $\mathbb{F}_q$.*

Under this heuristic, and recalling Definition 3, we have that the average number of matrices $\widetilde{\mathbf{Q}}$ one has to test, before a valid matrix is found, corresponds to $1/\gamma$. Notice that the heuristic is here employed since we consider that, for each choice of $\widetilde{\mathbf{Q}}$, $\widetilde{\mathbf{G}}$ behaves as a uniformly random matrix. For each $\widetilde{\mathbf{Q}}$ the prover executes RREF^* and then computes CF. Let T_{RREF^*} and T_{CF} be the costs of these functions, respectively; then, computing the commitment comes with cost $\frac{T_{\mathsf{RREF}^*}+T_{\mathsf{CF}}}{\gamma}$. As we have already seen, γ is in practice very high, so that $\frac{1}{\gamma} \approx 1$: the first choice of $\widetilde{\mathbf{Q}}$ is successful with overwhelming probability. Computing the response takes a much smaller time so, for simplicity, we do not consider it. Analogously, verification is predominated by performing Gaussian elimination and then computing the canonical form. So, on the verifier's side, the cost can be estimated as $T_{\mathsf{RREF}^*} + T_{\mathsf{CF}}$.

Whenever T_{CF} has a cost which is less than or, at the very least, comparable with T_{RREF^*}, the use of canonical forms does not lead to significant computational overhead. Indeed, as it is well known, a crude but realistic estimate for T_{RREF^*} is $O\big(n^3\big)$ field operations. As we have already seen in Section 4, it is possible to define canonical forms whose time complexity is much better than or comparable with that of T_{RREF^*}. Indeed, among the functions we have defined, the most time consuming one is that for Case 5, taking $O(dn^3)$ field operations in the worst case but $O(n^3)$ operations on average.

6 Hardness Analysis and Implications

In this section we provide strong evidence that the new formulation of code equivalence, using canonical forms, still leads to a hard problem. Namely, we prove that there exists a polynomial-time reduction from code equivalence to Problem 2, given that canonical forms can be computed in polynomial time. If the considered canonical forms have $\gamma = 1$ (i.e., they never fail), we obtain that the reduction works for every code equivalence instance. When instead $\gamma < 1$ we prove that, under Heuristic 1, the reduction is successful for all but a negligible portion of codes given that γ is large enough (say, it does not decrease exponentially with n). As we have already shown, we know about canonical forms which exist with overwhelming probability,

given that q is large enough. It follow that the reduction is meaningful for all such code equivalence instances and, in particular, for those which are relevant for cryptographic applications.

The reduction in the other way is trivial (we briefly sketch it, even though it has already been shown implicitly in the protocol of Figure 2). In practice, this means that the code equivalence problem with canonical forms is as hard as in its traditional formulation.

We furthermore show that the reduction may be used to mount a practical attack on code equivalence. When instantiated with canonical forms that are efficiently computable (as those in Section 4), we propose a collision search strategy that, when q is large enough, may even be faster than state-of-the-art attacks.

For all the reductions we provide in the paper, we consider the LEP version of code equivalence. Moreover, we consider the case in which $F = D_k \times S_k \times D_{n-k} \times S_{n-k}$, which is the most general case. All other cases can be treated in an analogous way.

6.1 Reductions between LEP and CF-LEP

Showing that CF-LEP reduces to LEP is trivial and has already been done, implicitly, in the description of the CF-LESS Sigma protocol. Namely, for a pair $\mathbf{G}, \mathbf{G}'$ for which a solution $\mathbf{Q} = \mathbf{PD}$ is known, it would be enough to continue sampling matrices $\mathbf{P}_{\text{is}} \in S_{k,n}$ until $\mathsf{CF}(\mathbf{G} \cdot \mathbf{P}_{\text{is}}) \neq \perp$. Then, from the knowledge of $\mathbf{P}$, one can easily find $\mathbf{P}'_{\text{is}}$ such that $\mathsf{CF}(\mathbf{G} \cdot \mathbf{P}_{\text{is}}) = \mathsf{CF}(\mathbf{G}' \cdot \mathbf{P}'_{\text{is}})$.

The other direction, i.e., showing that LEP reduces to CF-LEP is more interesting. The way to map a CF-LEP solution into a LEP solution is described in Algorithm 1; its correctness, together with the probability that the reduction succeeds, is detailed in the next Proposition.

Proposition 5. *Let $F = D_k \times S_k \times D_{n-k} \times S_{n-k}$ and CF be a canonical form for F, computable in polynomial time. If $(\mathbf{G}, \mathbf{G}')$ admits a solution for CF-LEP, then a solution for LEP, on input $(\mathbf{G}, \mathbf{G}')$,can be found in polynomial time.*

Proof. Let us denote by $\mathbf{P}_{\text{is}}, \mathbf{P}'_{\text{is}} \in S_{k,n}$ the solution for the CF-LEP instance $(\mathbf{G}, \mathbf{G}')$. Now, consider Algorithm 1, on input $\mathbf{G}, \mathbf{G}'$ and

Algorithm 1: Building LEP solution from CF-LEP solution

Data: $F = D_k \times S_k \times D_{n-k} \times S_{n-k}$, canonical form function $\mathsf{CF} : \mathbb{F}_q^{k\times(n-k)} \mapsto \{\{\bot\} \cup \mathbb{F}_q^{k\times(n-k)}\}$
Input: matrices $\mathbf{G}, \mathbf{G}' \in \mathbb{F}_q^{k\times n}$, solution $\mathbf{P}_{\text{is}}, \mathbf{P}'_{\text{is}}$ for IS-LEP
Output: solution $\mathbf{S} \in \mathrm{GL}_k$, $\mathbf{Q} \in M_n$ for LEP

1 Compute $\mathsf{RREF}(\mathbf{G}\cdot\mathbf{P}_{\text{is}}) = \big(\mathbf{I}_k \mid \mathbf{A}\big) = \mathbf{U}\cdot\mathbf{G}\cdot\mathbf{P}_{\text{is}}$;
2 Compute $\mathsf{RREF}(\mathbf{G}'\cdot\mathbf{P}'_{\text{is}}) = \big(\mathbf{I}_k \mid \mathbf{A}'\big) = \mathbf{U}'\cdot\mathbf{G}'\cdot\mathbf{P}'_{\text{is}}$;
3 Compute $\mathbf{B} = \mathsf{CF}(\mathbf{A}) = \mathbf{P}_r\cdot\mathbf{D}_r\cdot\mathbf{A}\cdot\mathbf{P}_c\cdot\mathbf{D}_c$;
4 Compute $\mathbf{B}' = \mathsf{CF}(\mathbf{A}') = \mathbf{P}'_r\cdot\mathbf{D}'_r\cdot\mathbf{A}'\cdot\mathbf{P}'_c\cdot\mathbf{D}'_c$;
5 Set $\widetilde{\mathbf{P}}_r\cdot\widetilde{\mathbf{D}}_r = (\mathbf{P}'_r\cdot\mathbf{D}'_r)^{-1}\cdot\mathbf{P}_r\cdot\mathbf{D}_r$;
6 Set $\widetilde{\mathbf{P}}_c\cdot\widetilde{\mathbf{D}}_c = \mathbf{P}_c\cdot\mathbf{D}_c\cdot(\mathbf{P}'_c\cdot\mathbf{D}'_c)^{-1}$;
7 Compute $\mathbf{Q} = \mathbf{P}_{\text{is}}\cdot\begin{bmatrix}\widetilde{\mathbf{P}}_r\cdot\widetilde{\mathbf{D}}_r & \mathbf{0}\\ \mathbf{0} & \widetilde{\mathbf{P}}_c\cdot\widetilde{\mathbf{D}}_c\end{bmatrix}\cdot\mathbf{P}'^{-1}_{\text{is}}$;
8 Compute $\mathbf{S} = \mathbf{U}'^{-1}\cdot(\widetilde{\mathbf{P}}_r\cdot\widetilde{\mathbf{D}}_r)^{-1}\cdot\mathbf{U}$;
9 **return** $\mathbf{S}, \mathbf{Q}$

$\mathbf{P}_{\text{is}}, \mathbf{P}_{\text{is}}$. The algorithm obviously takes polynomial time, since all it does is computing canonical forms (which takes polynomial time by hypothesis) and matrix multiplications/inversions. We only need to show it is correct, i.e., that the returned $\mathbf{S}$, $\mathbf{Q}$ indeed solves LEP. Since $\mathbf{P}_{\text{is}}, \mathbf{P}'_{\text{is}}$ is a solution for CF-LEP, this means that $\mathsf{CF}(\mathbf{G}\cdot\mathbf{P}_{\text{is}}) = \mathsf{CF}(\mathbf{G}\cdot\mathbf{P}'_{\text{is}})$. Recalling how canonical forms are defined, this is equivalent to first computing RREF and then computing CF on the non systematic part. Let $\mathbf{U}$ be the matrix that brings $\mathbf{G}\cdot\mathbf{P}_{\text{is}}$ into RREF form, that is, $\mathbf{U}\cdot\mathbf{G}\cdot\mathbf{P}_{\text{is}} = (\mathbf{I}_k \mid \mathbf{A})$. So $\mathsf{CF}\big(\mathbf{G}\cdot\mathbf{P}_{\text{is}}) = \mathsf{CF}(\mathbf{I}_k \mid \mathsf{CF}(\mathbf{A})\big)$ and, by definition of canonical forms,

$$\mathsf{CF}(\mathbf{A}) = \mathbf{P}_r\cdot\mathbf{D}_r\cdot\mathbf{A}\cdot\mathbf{P}_c\cdot\mathbf{D}_c.$$

We use analogous notation for $\mathbf{G}'$ (notice that we are following the notation of Algorithm 1), and get $\mathbf{U}'\cdot\mathbf{G}'\cdot\mathbf{P}'_{\text{is}} = (\mathbf{I}_k \mid \mathbf{A}')$ and

$$\mathsf{CF}(\mathbf{A}') = \mathbf{P}'_r\cdot\mathbf{D}'_r\cdot\mathbf{A}'\cdot\mathbf{P}'_c\cdot\mathbf{D}'_c.$$

Since $\mathsf{CF}(\mathbf{G}\cdot\mathbf{P}_{\text{is}}) = \mathsf{CF}(\mathbf{G}'\cdot\mathbf{P}'_{\text{is}})$, we have $\mathsf{CF}(\mathbf{A}) = \mathsf{CF}(\mathbf{A}')$, i.e.,

$$\mathbf{A}' = \underbrace{(\mathbf{P}'_r\cdot\mathbf{D}'_r)^{-1}\cdot\mathbf{P}_r\cdot\mathbf{D}_r}_{\widetilde{\mathbf{P}}_r\cdot\widetilde{\mathbf{D}}_r}\cdot\mathbf{A}\cdot\underbrace{\mathbf{P}_r\cdot\mathbf{D}_r\cdot(\mathbf{P}'_c\cdot\mathbf{D}'_c)^{-1}}_{\widetilde{\mathbf{P}}_c\cdot\widetilde{\mathbf{D}}_c}.$$

So, we can write

$$\mathbf{U}' \cdot \mathbf{G}' \cdot \mathbf{P}'_{\text{is}} = (\mathbf{I}_k \mid \mathbf{A}') = (\widetilde{\mathbf{P}}_r \cdot \widetilde{\mathbf{D}}_r)^{-1} \cdot \underbrace{(\mathbf{I}_k \mid \mathbf{A})}_{\mathbf{U}\cdot\mathbf{G}\cdot\mathbf{P}_{\text{is}}} \cdot \begin{bmatrix} \widetilde{\mathbf{P}}_r\widetilde{\mathbf{D}}_r & \mathbf{0} \\ \mathbf{0} & \widetilde{\mathbf{P}}_c\widetilde{\mathbf{D}}_c \end{bmatrix}$$
$$= (\widetilde{\mathbf{P}}_r \cdot \widetilde{\mathbf{D}}_r)^{-1} \cdot \mathbf{U} \cdot \mathbf{G} \cdot \mathbf{P}_{\text{is}} \cdot \begin{bmatrix} \widetilde{\mathbf{P}}_r\widetilde{\mathbf{D}}_r & \mathbf{0} \\ \mathbf{0} & \widetilde{\mathbf{P}}_c\widetilde{\mathbf{D}}_c \end{bmatrix}.$$

From the above, with simple algebraic manipulations, we obtain the expressions for $\mathbf{S}$ and $\mathbf{Q}$. □

We now comment on how many LEP instances admit also a solution for CF-LEP. To this end, we consider that an arbitrary LEP instance $(\mathbf{G}, \mathbf{G}')$ indeed admits a solution for CF-LEP in case there exists at least one matrix $\mathbf{P}_{\text{is}} \in S_{k,n}$ for which $\mathsf{CF}(\mathbf{G} \cdot \mathbf{P}_{\text{is}}) \neq \perp$. This corresponds to the complementary of the probability that, for all possible RREF forms for $\mathbf{G}$ and $\mathbf{G}'$, the canonical form is not defined. Whenever $\gamma = 1$, then canonical forms can always be defined and the above reduction applies to any LEP instance.

If instead $\gamma < 1$, then we can rely on Heuristic 1. From [19, Section 2], we know that a a random $k \times k$ matrix is non singular with probability ζ which is at least $1 - 1/q - 1/q^2$. So, for any set $J \subseteq \{1, \cdots, k\}$, ζ corresponds to the probability that J is an information set. According to the heuristic, after RREF, the non systematic part behaves as a uniformly random $k \times (n-k)$ matrix over $\mathbb{F}_q$, so it admits a canonical form with probability γ. Consequently, the probability that canonical forms cannot be defined for all sets J is $(1 - \zeta\gamma)^{\binom{n}{k}}$. Taking the the logarithm of this quantity and considering that $\log_2(x) \leq \frac{1}{\ln(2)}(x-1)$ for all positive $x \in \mathbb{R}$, we further get

$$\log_2 (1 - \zeta\gamma)^{\binom{n}{k}} \leq \binom{n}{k} \frac{(1 - \zeta\gamma) - 1}{\ln(2)} = -\binom{n}{k} \frac{\zeta\gamma}{\ln(2)}.$$

Thus, the number of LEP instances for which a solution for CF-LEP cannot be found is less than $2^{-\binom{n}{k}\frac{\zeta\gamma}{\ln(2)}} = 2^{-\psi(n,k,q)}$. In practice, this always close to 0: $\psi(n,k,q) \geq 0$, $\binom{n}{k}$ is exponential in n (when $k = Rn$ with R constant) and ζ is lower bounded by a constant. Thus, $\phi(n,k,q)$ can be close to 0 (so that $2^{-\psi(n,k,q))}$ is large) only

when γ is extremely small, say, $\gamma = 2^{-c \cdot n}$ for some positive $c \geq 0$. As we have seen, for the codes we consider in this paper, γ is always very large (almost 1), so that $2^{-\psi(n,k,q)}$ is negligible.

Remember that, for all canonical forms we have defined, the success probability gets smaller when q gets lower. In such a regime, there may exist different ways to define canonical forms with sufficiently large success probability. Finding canonical forms that work even when q is smaller would enlarge the range of code equivalence instances for which the reduction is meaningful; we leave this as an interesting open question.

6.2 Canonical Forms as a Solver for LEP

We now show how the reduction in Algorithm 1 can be used to mount a practical attack on code equivalence. Again, we focus on the case of LEP but the attack obviously works also when considering PEP. The core of our proposed procedure is shown in Algorithm 2. Essentially, the procedure first solves CF-LEP using a meet-in-the-middle strategy; then, it calls the reduction in Algorithm 2 to reconstruct the equivalence between $\mathbf{G}$ and $\mathbf{G}'$.

The analysis of the resulting time complexity is very simple. First, because of the birthday paradox, the algorithm has constant success probability which is approximately 1/2. For each candidate $\mathbf{P}_{\text{is}}$ (resp., $\mathbf{P}'_{\text{is}}$), the probability that it corresponds to a computable canonical form is $\zeta\gamma$ which, as we have already seen, is expected to be a constant (with value close to 1). Finally, we have shown that canonical forms can be computed in polynomial time. Indicating with $h(R) = -R\log_2(R) - (1-R)\log_2(1-R)$ the binary entropy function, we get an asymptotic time complexity of

$$O\left(|\mathcal{L}| + |\mathcal{L}'|\right) = O\left(\sqrt{\binom{n}{k}}\right) = 2^{n \cdot h(R) \cdot \left(1 + o(1)\right)}.$$

In practice, we can think of the algorithm as a clever way to brute force CF-LEP. Note that, for LEP, a brute force attack (using with the very same meet-in-the-middle approach) would take time $O\left(\sqrt{n!(q-1)^n}\right)$, which is super-exponential in n.

Algorithm 2: Solving code equivalence via canonical forms

Input: matrices $\mathbf{G}, \mathbf{G}' \in \mathbb{F}_q^{k \times n}$
Output: equivalence between $\mathbf{G}$ and $\mathbf{G}'$, or failure

Set $\mathcal{L} = \varnothing$, $\mathcal{L}' = \varnothing$;
// Populate first list
while $|\mathcal{L}| < \sqrt{\binom{n}{k}}$ **do**
    Sample $\mathbf{P}_{\text{is}} \xleftarrow{\$} S_{k,n}$;
    if $\mathbf{G} \cdot \mathbf{P}_{\text{is}}$ admits information set $\{1, \cdots, k\}$ **then**
        Compute $\mathbf{B} = \mathsf{CF}(\mathbf{G} \cdot \mathbf{P}_{\text{is}})$;
        if $\mathbf{B} \neq \perp$ **then**
            Add $\left(\mathbf{P}_{\text{is}}, \mathbf{B}\right)$ to $\mathcal{L}$;
// Populate second list
while $|\mathcal{L}'| < \sqrt{\binom{n}{k}}$ **do**
    Sample $\mathbf{P}'_{\text{is}} \xleftarrow{\$} S_{k,n}$;
    if $\mathbf{G}' \cdot \mathbf{P}'_{\text{is}}$ admits information set $\{1, \cdots, k\}$ **then**
        Compute $\mathbf{B}' = \mathsf{CF}(\mathbf{G}' \cdot \mathbf{P}'_{\text{is}})$;
        if $\mathbf{B}' \neq \perp$ **then**
            Add $\left(\mathbf{P}'_{\text{is}}, \mathbf{B}'\right)$ to $\mathcal{L}'$;
// Find solution for CF-LEP, then reconstruct the equivalence
Search for collisions, i.e., pairs $(\mathbf{P}_{\text{is}}, \mathbf{B}) \in \mathcal{L}$, $(\mathbf{P}'_{\text{is}}, \mathbf{B}') \in \mathcal{L}'$ such that $\mathbf{B} = \mathbf{B}'$;
If a collision is found, call Algorithm 1 on input $\mathbf{G}, \mathbf{G}'$ and $\mathbf{P}_{\text{is}}$, $\mathbf{P}'_{\text{is}}$

All known attacks on LEP have complexity which grows with q; however, the complexity resulting from Algorithm 2 does not depend on this factor[11]. Appendix A briefly compares Algorithm 2 with other known attacks. The analysis is only qualitative, since we use a very rough estimate for the costs of the algorithms based on short codewords [7, 8, 21]. Also, we are assuming that γ is a (large) constant, but this may require more precise evaluations (especially since, for growing n, the success probabilities may change). Yet, our preliminary results show that, when q is large enough, Algorithm 1 may significantly outperform state-of-the-art attacks. It is worth mentioning here, once more, that this is not the regime of interest for LESS, and thus such results do not affect the claimed security levels.

[11] For concrete estimates, one would need to consider this since ζ, γ, and the cost of computing canonical forms all depend on q. Yet, this consideration would only lead to constant or polynomial factors, which disappear in the asymptotic regime.

7 Concrete Instantiations

In this section, we discuss concrete instantiations of LESS using our technique. It is worth noting that canonical forms allow us achieve almost-optimal sizes. Indeed, we have $n \approx 2\lambda$ and our newly-proposed attack technique shows that we have attacks with asymptotic complexity $O\left(2^{\frac{n}{2} \cdot h(1/2)}\right) = O\left(2^{\frac{n}{2}}\right)$. Thus, whenever the code rate is $1/2$, the codes have sizes which are close to being optimal (if polynomial factors are omitted). Given that $n \approx 2\lambda$, we are sending responses that are as large as commitments and cannot use smaller values for n.

NIST Cat.	**Type**	**Code Params** n	k	q	**Prot. Params** s	t	w	**Attack Factor**	**Case**	**pk** (B)	**sig** (B)
1	Mono	252	126	127	2	247	30	123.84	2	13939	8624
	Mono								4		5789
	Mono								5		2481
	Perm								3		2481
3	Mono	400	200	127	2	759	33	197.67	2	35074	17208
	Mono								4		11433
	Mono								5		5658
	Perm								3		5658
5	Mono	548	274	127	2	1352	40	271.56	2	65792	30586
	Mono								4		19626
	Mono								5		10036
	Perm								3		10036

Table 2. Potential parameter sets with $s = 2$ for CF-LESS. All sizes in bytes (B).

Table 2 shows some potential parameter sets for CF-LESS, along with the parameter sets with $s = 2$ of LESS; the latter are reported in the bottom row of each cell. The main purpose of this table is to illustrate the impact of our technique; therefore, we report sizes corresponding to the various choices of F defined in our work, indicating which one was considered in the column "Case". The column "Attack Factor" indicates the largest factor in the cost of of the attack in Section 6.2, estimated as $\log_2 \sqrt{\binom{n}{k}}$. Note that the number of bit operations taken by the attack is more than $\sqrt{\binom{n}{k}}$, as there are other nontrivial factors such as T_{CF}. Giving the exact bit operation counts is out of the scope of this paper.

The signature sizes in the column "sig" are computed as

$$w \cdot \lceil \mathsf{isobits}/8 \rceil + \Lambda(t,w) \cdot \ell_{\text{tree\_seed}} + \ell_{\text{salt}} + \ell_{\text{digest}}.$$

The value of $\mathsf{isobits}$ has been computed in Table 1. $\Lambda(t,w)$ indicates the number of seeds (in the tree) that we need. We define $\Lambda(t,w)$ as $2^{\lceil \log_2 w \rceil} + w \cdot (\lceil \log_2 t \rceil - \lceil \log_2 w \rceil - 1)$, as in [14, 20]. $\ell_{\text{tree\_seed}}$, ℓ_{salt} and ℓ_{digest} stand for the respective lengths of seeds, salt and digest. These values have been specified in [3, Table 2].

Of course, one can achieve even smaller signature sizes by increasing s, at the cost of larger public keys. We report these below.

NIST Cat.	**Type**	**Code Params** n	k	q	**Prot. Params** s	t	w	**Attack Factor**	**Case**	**pk** (B)	**sig** (B)
1	Mono	252	126	127	4	244	20	123.84	2	41785	5941
	Mono								4		4051
	Mono								5		1846
	Perm								3		1846
3	Mono	400	200	127	4	895	24	197.67	2	105174	12768
	Mono								4		8568
	Mono								5		4368
	Perm								3		4368
5	Mono	548	274	127	4	907	34	271.56	2	197312	25237
	Mono								4		15921
	Mono								5		7769
	Perm								3		7769

Table 3. Potential parameter sets with $s = 4$ for CF-LESS. All sizes in bytes (B).

Failure Probability. We have experimentally confirmed that the failure rate is reasonably low for each parameter set in Table 2. To this end, we generated 2^{20} random matrices in $\mathbb{F}_q^{k \times (n-k)}$ for each parameter set (for Case 4 and 5 we set $d = 5$) and were always able to derive the corresponding canonical forms. Even better, in Appendix B we show that the failure probability is below 2^{-165} when the algorithm in Case 3 is used. With such low failure probabilities, we conclude that the reduction of Proposition 5 is efficient, and the resulting overhead for signing is consequently quite small.

Comparison with Other Schemes. Table 4 shows public key and signature sizes of some other post-quantum signature schemes.

Parameter Set	pk	sig	NIST Cat.	Reference
SPHINCS+-128s	32	7856	1	[2]
SPHINCS+-192s	32	16224	3	
SPHINCS+-256s	32	29792	5	
MEDS-9923	9923	9896	1	[16]
MEDS-41711	41711	41080	3	
MEDS-134180	134180	132528	5	
uov-Ip	43576	128	1	[10]
uov-Is	66576	96	1	
uov-III	189232	200	3	
uov-V	446992	260	5	
MiRitH-Ia	129	5673	1	[1]
MiRitH-IIIa	205	12440	3	
MiRitH-Va	253	21795	5	

Table 4. Public key and signature sizes of other post-quantum signature schemes.

Compared to MEDS, which is similar in design, CF-LESS has comparable public key sizes but much smaller signatures. SPHINCS+ and MPC-in-the-head schemes like MiRitH, have characteristically very small public keys, but their signature sizes are much larger. CF-LESS can be an attractive option, then, in applications where signature size has more impact than public key size. On the other hand, when compared to schemes such as UOV, CF-LESS has smaller public keys and larger signatures. Still, we remark that CF-LESS performs better if the sum of public key and signature is considered.

There are other post-quantum schemes featuring both smaller public keys and smaller signatures than CF-LESS, such as MAYO [9] and SQIsign [15]. The underlying problem in MAYO is still quite young, so it probably requires more study to gain confidence. On the other hand, SQISign's reported performance numbers are still far from practical (e.g. very slow signing and key generation, in the order of billions of cycles). Overall, our scheme is be well-positioned in the post-quantum signature ecosystem, especially among code-based signatures, where we exhibit the smallest signature size, except for hash-and-sign schemes such as Wave [4], which are however hindered by huge public keys (e.g. 3.6 MB at category 1).

References

1. Adj, G., Rivera-Zamarripa, L., Verbel, J., Bellini, E., Barbero, S., Esser, A., Sanna, C., Zweydinger, F.: MiRitH (MinRank in the Head) (2023), https://pqc-mirith.org/assets/downloads/mirith_specifications_v1.0.0.pdf
2. Aumasson, J.P., Bernstein, D.J., Beullens, W., Dobraunig, C., Eichlseder, M., Fluhrer, S., Gazdag, S.L., Hülsing, A., Kampanakis, P., Kölbl, S., Lange, T., Lauridsen, M.M., Mendel, F., Niederhagen, R., Rechberger, C., Rijneveld, J., Schwabe, P., Westerbaan, B.: SPHINCS$^+$ (2022), https://sphincs.org/data/sphincs+-r3.1-specification.pdf
3. Baldi, M., Barengh, A., Beckwith, L., Biasse, J.F., Esser, A., Gaj, K., Mohajerani, K., Pelosi, G., Persichetti, E., Saarinen, M.J.O., Santini, P., Wallace, R.: LESS: Linear equivalence signature scheme (2023), https://www.less-project.com/LESS-2023-08-18.pdf
4. Banegas, G., Karpman, P., Carrier, K., Loyer, J., Chailloux, A., Niederhagen, R., Couvreur, A., Sendrier, N., Debris-Alazard, T., Smith, B., Gaborit, P., Tillich, J.P.: Wave (2017), https://wave-sign.org/
5. Bardet, M., Otmani, A., Saeed-Taha, M.: Permutation code equivalence is not harder than graph isomorphism when hulls are trivial. In: 2019 IEEE International Symposium on Information Theory (ISIT). pp. 2464–2468. IEEE (2019). https://doi.org/10.1109/ISIT.2019.8849855
6. Barenghi, A., Biasse, J.F., Persichetti, E., Santini, P.: LESS-FM: fine-tuning signatures from the code equivalence problem. In: Post-Quantum Cryptography: 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20–22, 2021, Proceedings 12. pp. 23–43. Springer (2021). https://doi.org/10.1007/978-3-030-81293-5_2
7. 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
8. Beullens, W.: Not enough less: An improved algorithm for solving code equivalence problems over $\mathbb{F}_q$. In: Selected Areas in Cryptography: 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21-23, 2020, Revised Selected Papers. pp. 387–403. Springer (2021). https://doi.org/10.1007/978-3-030-81652-0_15
9. Beullens, W., Campos, F., Celi, S., Hess, B., Kannwischer, M.J.: MAYO (2023), https://pqmayo.org/assets/specs/mayo.pdf
10. Beullens, W., Chen, M.S., Ding, J., Gong, B., Kannwischer, M.J., Patarin, J., Peng, B.Y., Schmidt, D., Shih, C.J., Tao, C., Yang, B.Y.: UOV: Unbalanced Oil and Vinegar (2023), https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/UOV-spec-web.pdf
11. Beullens, W., Katsumata, S., Pintore, F.: Calamari and Falafl: logarithmic (linkable) ring signatures from isogenies and lattices. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 464–492. Springer (2020). https://doi.org/10.1007/978-3-030-64834-3_16
12. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-Fish: efficient isogeny based signatures through class group computations. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 227–247. Springer (2019). https://doi.org/10.1007/978-3-030-34578-5_9
13. Biasse, J.F., Micheli, G., Persichetti, E., Santini, P.: LESS is more: code-based signatures without syndromes. In: Progress in Cryptology - AFRICACRYPT 2020:

12th International Conference on Cryptology in Africa, Cairo, Egypt, July 20–22, 2020, Proceedings 12. pp. 45–65. Springer (2020). https://doi.org/10.1007/978-3-030-51938-4_3
14. Boyar, J., Erfurth, S., Larsen, K.S., Niederhagen, R.: Quotable signatures for authenticating shared quotes. In: Progress in Cryptology – LATINCRYPT 2023. pp. 273–292. Springer (2023). https://doi.org/10.1007/978-3-031-44469-2_14, https://arxiv.org/pdf/2212.10963.pdf
15. 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 (2023), https://sqisign.org/spec/sqisign-20230601.pdf
16. Chou, T., Niederhagen, R., Persichetti, E., Ran, L., Randrianarisoa, T.H., Reijnders, K., Samardjiska, S., Trimoska, M.: MEDS Matrix Equivalence Digital Signature (2023), https://www.meds-pqc.org/spec/MEDS-2023-07-26.pdf
17. Feo, L.D., Galbraith, S.: SeaSign: Compact isogeny signatures from class group actions. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. Lecture Notes in Computer Science, vol. 11478, pp. 759–789. Springer (2019). https://doi.org/10.1007/978-3-030-17659-4_26
18. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Advances in Cryptology — CRYPTO' 86. pp. 186–194. Springer (1986). https://doi.org/10.1007/3-540-47721-7_12
19. FULMAN, J., GOLDSTEIN, L.: Stein's method and the rank distribution of random matrices over finite fields. The Annals of Probability **43**(3), 1274–1314 (2015). https://doi.org/10.1214/13-AOP889
20. Gueron, S., Persichetti, E., Santini, P.: Designing a practical code-based signature scheme from zero-knowledge proofs with trusted setup. Cryptography **6**(1), 5 (2022). https://doi.org/10.3390/cryptography6010005
21. 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
22. NIST: Call for Additional Digital Signature Schemes for the Post-Quantum Cryptography Standardization Process (2023), https://csrc.nist.gov/projects/pqc-dig-sig/standardization/call-for-proposals
23. Persichetti, E., Santini, P.: A new formulation of the linear equivalence problem and shorter less signatures (2023), https://eprint.iacr.org/2023/847
24. 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
25. Sendrier, N., Simos, D.E.: The hardness of code equivalence over $\mathbb{F}_q$ and its application to code-based cryptography. In: Post-Quantum Cryptography: 5th International Workshop, PQCrypto 2013, Limoges, France, June 4-7, 2013. Proceedings 5. pp. 203–216. Springer (2013). https://doi.org/10.1007/978-3-642-38616-9_14

A Comparison with Other Solvers

In this section, we consider various solvers for the code equivalence problem, and compare their running time with the one of our algorithm from Section 6.2.

SSA, [24]: this algorithm can efficiently solve PEP when the hull of the considered codes is small. However, the attack takes exponential time when the hull is large, as is the case for self-orthogonal codes (i.e. contained in their dual); in such a case, it has time complexity $T_{SSA} = O(q^k) = O(2^{Rn \cdot \log_2(q)})$. Thanks to a reduction in [25], SSA can also be used to solve LEP; however, whenever $q \geq 5$, the reduction maps any code into a self-orthogonal code with dimension k (so, it has time complexity $O(q^k)$).

BOS, [5]: this algorithm reduces PEP to graph isomorphism. While the technique is efficient for codes whose hull is either trivial or has small dimension, it yields super-exponential running time $T_{\text{BOS}} = O(n^{Rn})$ when self-orthogonal codes are considered.

Leon, *Beullens*, *BBPS*, [7, 8, 21]: each of these algorithms exhibits some peculiar aspects and may work only in certain regimes. For instance, while Leon's algorithm works regardless of q, Beullens' algorithm is very likely to fail when q is too small. Both of these algorithms can solve both PEP and LEP, while the BBPS algorithm improves upon Beullens' LEP algorithm by exploiting short codewords instead of subcodes. A precise estimate for the time complexity of each of these algorithms would depend on several factors which are sometimes hard to take into account. For instance, Leon requires to find all codewords whose weight is not greater than some value w which (heuristically) can be set slightly larger than the minimum distance: however, to the best of our knowledge, a formula to set w a priori is not known. In any case, these three algorithms follows a common principle, since they do not depend on the hull dimension and require to find a sufficiently large number of short codewords (or subcodes). For the sake of simplicity, for these three algorithms we consider the cost of finding a unique low-weight codeword us-

ing Prange's algorithm[12][13] Hence, for these algorithms we consider a time complexity given by

$$T = O\left(2^{\tau_{\text{Prange}}(R,q)\left(1+o(1)\right)}\right),$$

where

$$\tau_{\text{Prange}}(R,q) = h_2(R) - \left(1 - h_q^{-1}(1-R)\right) \cdot h_2\left(\frac{R}{1-h_q^{-1}(1-R)}\right),$$

and h_q denotes the q-ary entropy function.

We are now ready to compare the above algorithms with Algorithm 2; to this end, consider Figure 3. We are considering code equivalence instances for which both SSA and BOS are no efficient (i.e., PEP with self-orthogonal codes or LEP with $q \geq 5$). Note that SSA and BOS have been omitted from the comparison since their performance would have not been competitive: BOS runs in time which is super-exponential in the code length n while SSA is sometimes faster than our algorithm only if $q \leq 7$. We see that, when q is small, our algorithm is significantly slower than those based on codeword finding. Instead, when q grows, our algorithm becomes much more competitive and becomes faster then Prange.

We remark that this analysis holds given that efficiently computable canonical forms are considered. The ones introduced in this paper work whenever q is large enough, while they may yield a small success probability when q gets lower: this may make our attack slower. We have not analyzed how these canonical forms work when q gets lower; we see this, and even the question of whether new canonical forms may exist, as interesting research perspectives.

[12] The choice of Prange's ISD is meaningful since, for large finite fields, modern algorithms such as Lee-Brickell and Stern seem to perform worse.

[13] Even though this provides only a very broad estimate of the actual time complexity, this allows us to compare with these algorithms in a simple and concise way. We point out that cryptanalysis is not the focus of this paper and the aim of this section is merely to show that canonical forms can be a useful tool not only for the design of cryptographic schemes, but also for the cryptanalysis of the code equivalence problem.

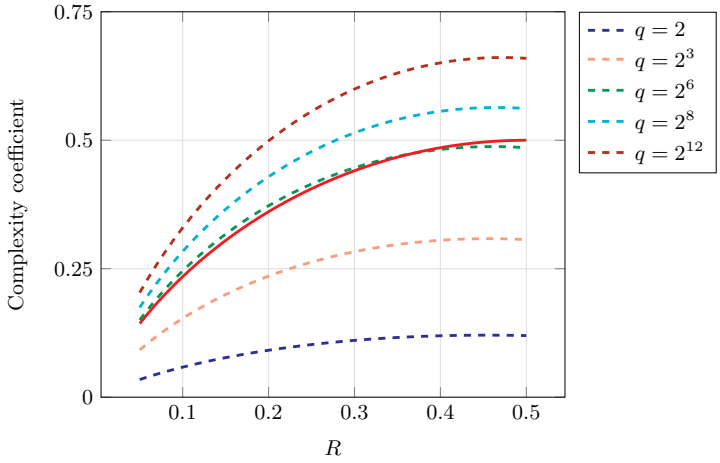

Fig. 3. Comparison between the complexity coefficients for Prange (dashed lines) and Algorithm 2 (continuous red line), as a function of the code rate.

Observe that the time complexity of Prange deteriorates quickly. This is due to the fact that, as q grows, the minimum distance approaches $n-k$ (since random codes meet the Singleton's bound with high probability). Hence, there is a unique information set which would result in a success for Prange's ISD: this is corroborated by the fact that $h_q^{-1}(1-R) \to 1-R$ as q grows and $\tau_{\text{Prange}}(R,q) \to h_2(R)$. Note that this complexity coefficient is twice the one which is achieved by our algorithm.

Asymptotic cost of Prange's ISD. A random code of length n and rate R has with overwhelming probability minimum distance $d = \delta n$, where $\delta = h_q^{-1}(1-R)$ (where h_q is the q-ary entropy function). The average number of iterations which are performed by the algorithm is

$$\frac{\binom{n}{k}}{\binom{n-d}{k}} = \frac{\binom{n}{Rn}}{\binom{n(1-\delta)}{Rn}} = 2^{n\cdot\left(h_2(R)-(1-\delta)\cdot h_2\left(\frac{R}{1-\delta}\right)\right)\left(1+o(1)\right)}.$$

The cost of each iteration is that of one Gaussian elimination: this is a polynomial term so we do not consider it. Then, for the algorithm we assume a complexity coefficient given by

$$\tau_{\text{Prange}}(R,q) = h_2(R) - (1-\delta)\cdot h_2\left(\frac{R}{1-\delta}\right).$$

B A Lower Bound on the Success Probability of the Canonical Form for Case 3

We derive a closed form, lower bound for the success probability of the canonical from Section 4, case 3, which is defined for $F = \{\mathbf{I}_k\} \times S_k \times \{\mathbf{I}_{n-k}\} \times S_{n-k}$.

Proposition 6. *For $\mathbf{A} \in \mathbb{F}_q^{k\times(n-k)}$ chosen uniformly at random, the canonical form defined as in Section 4, Case 3 exists with probability at least $\prod_{i=1}^{k-1} 1 - \frac{im}{q^{n-k}}$, where*

$$m = \begin{cases} (n-k)! & \text{if } n-k \leq q, \\ \frac{(n-k)!}{\left(v!\right)^{q(v+1)-(n-k)}\left((v+1)!\right)^{n-k-qv}} & \text{if } n-k > q, \end{cases}$$

where $v = \lfloor (n-k)/q \rfloor$.

Proof. We use $\mathbf{a}_i$ to indicate the i-th row of $\mathbf{A}$ and $\mathcal{S}(\mathbf{a}_i)$ to denote the set of vectors whose multiset is equal to that of $\mathbf{a}_i$. Notice that $\mathcal{S}(\mathbf{a}_i)$ contains all vectors that one can obtain by permuting the entries of $\mathbf{a}_i$. The probability that computational of canonical form does not fail can be lower bounded with a simple iterative reasoning.

Let us consider the first two rows of $\mathbf{A}$: regardless of $\mathbf{a}_1$, they will have different multisets if $\mathsf{multiset}(\mathbf{a}_2) \neq \mathsf{multiset}(\mathbf{a}_1)$. So, the probability that this pair of rows is valid is

$$\begin{aligned} \Pr\left[\{\mathbf{a}_1, \mathbf{a}_2\} \text{ is valid}\right] &= \sum_{\mathbf{a}_i \in \mathbb{F}_q^n} \Pr\left[\mathbf{a}_2 \text{ is valid} \mid \mathbf{a}_1\right] \cdot \Pr\left[\mathbf{a}_1\right] \\ &= \frac{1}{q^{n-k}} \sum_{\mathbf{a}_i \in \mathbb{F}_q^n} \left(1 - \frac{|\mathcal{S}(a_1)|}{q^{n-k}}\right), \end{aligned}$$

where $\Pr\left[\mathbf{a}_1\right]$ is the probability that the first row is equal to $\mathbf{a}_1$ and is equal to $q^{-(n-k)}$ for each $\mathbf{a}_1$ (since $\mathbf{A}$ is sampled according to the uniform distribution). Now, let m such that $|\mathcal{S}(\mathbf{a}_1)| \leq m$ for each possible $\mathbf{a}_1$: we get

$$\Pr\left[\{\mathbf{a}_1, \mathbf{a}_2\} \text{ is valid}\right] \geq \frac{1}{q^{n-k}} \sum_{\mathbf{a}_i \in \mathbb{F}_q^n} \left(1 - \frac{m}{q^{n-k}}\right) = 1 - \frac{m}{q^{n-k}}.$$

We now consider $\mathbf{a}_3$ and, with analogous reasoning, get that for any valid pair $\{\mathbf{a}_1, \mathbf{a}_2\}$, a new vector $\mathbf{a}_3$ is valid only if it does not belong to $\mathcal{S}(\mathbf{a}_1) \cup \mathcal{S}(\mathbf{a}_2)$. Using the upper bound m for both sets, we get that $\mathbf{a}_3$ is valid with probability at least $1 - \frac{2m}{q^{n-k}}$. If we iterate the reasoning up to the k-th row, we obtain the following probability:

$$\prod_{i=1}^{k-1} 1 - \frac{im}{q^{n-k}}.$$

Now we just need to derive useful values for m. To this end, we consider that, when $n-k \geq q$, then we can set $m = (n-k)!$: indeed, $|\mathcal{S}(a_1)| = (n-k)!$ holds only if $\mathbf{a}_1$ has all distinct entries while, otherwise $|\mathcal{S}(a_1)|$ contains fewer vectors. When $n - k > q$, we can refine the bound by taking into account that each $\mathbf{a}_i$ must necessarily have some repeated entries. The proof on how m is derived, in this case, is reported below. □

Maximum Number of Permutations for Vectors with Repeated Entries. We study the following problem: find the maximal value that $|\mathcal{S}(\mathbf{a})|$ can have, when $\mathbf{a}$ is a length-z vector over $\mathbb{F}_q$. Let ℓ_i denote the number of entries of $\mathbf{a}$ with value equal to $x_i \in \mathbb{F}_q$ (we are writing the field as $\{x_0 = 0, x_1 = 1, x_2, \cdots, x_{q-1}\}$); notice that it must be $\sum_{i=0}^{q-1} \ell_i = z$. The values ℓ_i allow us to take into account the number of permutations with repetitions, so that

$$|\mathcal{S}(\mathbf{a})| = \frac{z!}{\prod_{i=0}^{q-1} \ell_i!} = \frac{z!}{f(\ell_0, \ldots, \ell_{q-1})}.$$

Maximizing $|\mathcal{S}(\mathbf{a})|$ means minimizing $f(\ell_0, \ldots, \ell_{q-1})$: as we show next, this is achieved when all values ℓ_i are balanced, i.e., the difference between any pair of values ℓ_i, ℓ_j is not greater than 1.

Proposition 7. *For any $(\ell_0, \cdots, \ell_{q-1}) \in \mathbb{N}^q$ such that $\sum_{i=0}^{q-1} \ell_i = z$, it holds that*

$$f(\ell_0, \cdots, \ell_{q-1}) \geq (v!)^{q(v+1)-z} \left((v+1)!\right)^{z-qv},$$

where $v = \left\lfloor \frac{z}{q} \right\rfloor$.

Proof. The proof is crucially based on the simple observation that

$$\forall x, y \in \mathbb{N}, \text{ it holds } \quad y!x! > (y-1)!(x+1)! \text{ if } y - x > 1. \qquad (1)$$

Let us consider an arbitrary tuple $(\ell_0, \cdots, \ell_{q-1})$, summing to z, and assume there are two values ℓ_j, ℓ_u such that $\ell_j - \ell_u > 1$. Then, there exists a new tuple $(\ell'_0, \cdots, \ell'_{q-1})$ such that $\ell'_i = \ell_i$ for any $i \neq j, u$, $\ell'_j = \ell_j - 1$ and $\ell'_u = \ell_u + 1$. First, this configuration is valid since the sum of all the ℓ'_i is still equal to z. Also, because of (1), we have that

$$\frac{f(\ell_0, \cdots, \ell_{q-1})}{f(\ell'_0, \cdots, \ell'_{q-1})} = \frac{\prod_{i=0}^{q-1} \ell_i!}{\prod_{i=0}^{q-1} \ell'_i!} = \frac{\ell_j!\ell_u!}{\ell'_j!\ell'_u!} = \frac{\ell_j!\ell_u!}{(\ell_j - 1)!(\ell_u + 1)!} > 1.$$

We can iterate the procedure until we end up with a tuple where, for each pair of values, the difference is at most 1. This implies that there are only two possible values in the tuple, $v = \left\lfloor \frac{z}{q} \right\rfloor$ and $v + 1$. Let w denote the number of entries with value v: since it must be $vw + (q - w)(v + 1) = z$, we find $w = q(v + 1) - z$. So, the number of entries with value equal to $v + 1$ is $q - w = z - qv$. □

It follows that

$$\forall \mathbf{a} \in \mathbb{F}_q^z, \ |\mathcal{S}(\mathbf{a})| \leq \frac{z!}{\big(\lfloor z/q \rfloor !\big)^{q\left(\left\lfloor \frac{z}{q} \right\rfloor + 1\right) - z} \big((\lfloor z/q \rfloor + 1)!\big)^{z - q\left\lfloor \frac{z}{q} \right\rfloor}}.$$