Reducing Signature Size of Matrix-code-based Signature Schemes

Tung Chou[1], Ruben Niederhagen[1,2], Lars Ran[3], and Simona Samardjiska[3]

[1] Academia Sinica, Taipei, Taiwan
blueprint@crypto.tw
[2] University of Southern Denmark, Odense, Denmark
ruben@polycephaly.org
[3] Radboud University, Nijmegen, Netherlands
{lran,simonas}@cs.ru.nl

Abstract. This paper shows novel techniques to reduce the signature size of the code-based signature schemes MEDS and ALTEQ, by a large factor. For both schemes, the signature size is dominated by the responses for rounds with nonzero challenges, and we reduce the signature size by reducing the size of these responses. For MEDS, each of the responses consists of $m^2 + n^2$ field elements, while in our new protocol each response consists of only $2k$ (k is usually chosen to be close to m and n) field elements. For ALTEQ, each of the responses consists of n^2 field elements, while in our new protocol each response consists of about $\sqrt{2}n^{3/2}$ field elements. In both underlying Σ-protocols of the schemes, the prover generates a random isometry and sends the corresponding isometry to the verifier as the response. Instead of doing this, in our new protocols, the prover derives an isometry from some random code words and their presumed (full or partial) images. The prover sends the corresponding code words and images to the verifier as the response, so that the verifier can derive an isometry in the same way. Interestingly, it turns out that each response takes much fewer field elements to represent in this way.

Keywords: code-based cryptography · digital signature schemes · post-quantum cryptography

1 Introduction

In recent years, post-quantum cryptography has come into the spotlight of the cryptographic community as a result of several standardization efforts including that of the National Institute of Standards and Technology of the USA (NIST) [20]. With the goal of standardizing post-quantum key encapsulation mechanisms and digital signatures, NIST standardization process has spurred an enormous amount of new design and cryptanalytic ideas. We now evidently

This research has been partially supported by the Dutch government through the NWO grant OCNW.M.21.193 (ALPaQCa) and by the Taiwanese government through the NSTC grants 112-2634-F-001-001-MBK and 112-2222-E-0001-003.

have achieved great progress in understanding the security of new, but also relatively old proposals. In fact, some breakthrough cryptanalytic results against candidates [4,23] in the final round for standardization urged NIST to open an additional round for digital signatures [1] expecting to achieve more diversity in underlying hard problems and ratio between signature and key sizes. In this additional round, NIST indicated that they would like to select schemes with small signatures and fast verification that are not based on structured lattices. Immediate candidates that fit the description are multivariate signatures based on UOV [19], which inherently have very small signatures. The downside of these is that they typically have huge public keys and no guarantees about the security of the construction.

On the other end of the spectrum, lie the heavy but provably secure Fiat-Shamir signatures. In a course of a few years, thanks to generic huge improvements in signature size, they moved from extremely inefficient to reasonable candidates for standardization. There are now more than 12 candidates in the additional round based on the Fiat-Shamir paradigm. Three of these, MEDS [11], ALTEQ [22] and LESS [3] use the GMW Σ-protocol [17] by Goldreich, Micali and Wigderson, that was originally presented over the graph-isomorphism problem, but can be constructed from any hard equivalence problem. For example, MEDS uses the matrix code equivalence problem, in which the objects are matrix codes and the equivalences are two-sided bijective linear transformations. ALTEQ uses alternating trilinear form equivalence with equivalences again from the general linear group, but now acting on three "sides". Finally, LESS uses linear code equivalence where the objects are Hamming codes and the equivalences scaled permutations.

In all of these schemes the isometries are encoded in the signature and actually constitute most of it. Finding a compact representation of isometries thus directly influences the size of the signature. In this paper, our aim to encode isometries more efficiently while keeping the impact on the other performance metrics (public key size and computational performance) moderate.

1.1 Our Contribution

This paper shows novel techniques to reduce the signature size of the code-based signature schemes MEDS and ALTEQ, by a large factor. For both schemes, the signature size is dominated by the responses for rounds with nonzero challenges, and we reduce the signature size by reducing the size of these responses. For MEDS, each of these responses consists of $m^2 + n^2$ field elements, while in our new protocol each consists of only $2k$ (k is usually chosen to be close to m and n) field elements. For ALTEQ, each of these responses consists of n^2 field elements, while in our new protocol each consists of about $\sqrt{2}n^{3/2}$ field elements. Furthermore, using a similar technique, we can reduce the public key size by roughly $n(n - \sqrt{2n})$ field elements.

Concrete signature sizes can be found in in Table 1 and Table 3. For NIST security level I, the smallest signature size in the MEDS submission is 9896 bytes, while Table 1 shows a parameter set with 2886-byte signatures when our new

protocol is used. Also for NIST security level I, the smallest signature size of the ALTEQ submission is 9528 bytes, while Table 3 shows a parameter set with 3752-byte signatures when our new protocol is used. These smaller signature sizes are obtained without increasing the public key sizes.

1.2 Techniques

In the underlying Σ-protocols of both schemes, the prover generates a random isometry and sends the isometry, corresponding to the challenge, to the verifier as the response. Instead of doing this, in our new protocols, the prover derives an isometry from some random codewords and their presumed (full or partial) images. The prover sends the corresponding codewords and images to the verifier as the response, so that the verifier can derive the corresponding isometry in the same way. Each response takes much fewer field elements to represent in this way, as the codewords can be represented as their coordinates with respect to a basis of the code, and as the images do not even have to be sent, but can be simply considered as public data.

We take inspiration from the public key reduction technique introduced for MEDS in [11]. There, the fact that an isometry can be efficiently derived from a small number of matching codewords in the two codes is used in the key-generation process. Using it, it becomes possible to generate part of the public key from a public seed, and thus to reduce the overall size of the public key.

Although in this paper we only show new protocols for matrix-code-based signature schemes, our technique is a general approach for representation of isometries, and thus it might be efficiently applicable to other schemes as well. We leave this as future work.

1.3 Related Work: CF-LESS

Since LESS is constructed in a way similar to MEDS and ALTEQ, it seems reasonable to mention the recent paper [12], which is about reducing signature size of LESS. The technique of [12] looks quite different from ours. Roughly speaking, the main idea of [12] is to define an equivalence relation for the set of all possible codes and to set the commitment to be the canonical element in an equivalence class. In this way, each response does not need to contain the information for indicating which code in the equivalence class is the actual target, so the response size can be saved. In the Σ-protocol for the LESS submission [2], each response takes $k \cdot (\lceil \log_2 n \rceil + \lceil \log_2(q-1) \rceil)$ bits (where k is the code dimension, n is the code length, and q is the field size), while in the best Σ-protocol of [12], each response takes only n bits.

1.4 Structure of this Paper

Section 2 specifies notations that are useful for the remaining sections. Section 3 reviews how MEDS was designed. Section 4 presents our new Σ-protocols for MEDS. Section 5 reviews how ALTEQ was designed and presents our new Σ-protocols for ALTEQ.

2 Notations

We denote by $\mathsf{RS}(\mathbf{G})$ the row space of a matrix $\mathbf{G}$. We denote by $\mathsf{RREF}(\mathbf{G})$ reduced row echelon form of a matrix $\mathbf{G}$. $\mathsf{SF}(\mathbf{G})$ is defined to be $\mathsf{RREF}(\mathbf{G})$, if that happens to be of the form $(\mathbf{I} \mid \mathbf{H})$ (i.e., systematic form), or $\perp$ otherwise. We define $\mathsf{SF}^*(\mathbf{G})$ as follows.

$$\mathsf{SF}^*(\mathbf{G}) = \begin{cases} (\perp, \perp), & \text{if } \mathsf{SF}(\mathbf{G}) = \perp. \\ (\mathsf{SF}(\mathbf{G}), \mathbf{T}), \text{ such that } \mathsf{SF}(\mathbf{G}) = \mathbf{T} \cdot \mathbf{G}, & \text{otherwise.} \end{cases}$$

For any $\mathbf{v} \in \mathbb{F}_q^{mn}$, $\mathsf{matrix}(\mathbf{v})$ is defined as

$$\begin{bmatrix} \mathbf{v}_0 & \cdots & \mathbf{v}_{n-1} \\ \mathbf{v}_n & \cdots & \mathbf{v}_{2n-1} \\ \vdots & & \vdots \\ \mathbf{v}_{(m-1)n} & \cdots & \mathbf{v}_{mn-1} \end{bmatrix} \in \mathbb{F}_q^{m \times n}.$$

vector performs the inverse operation of matrix. Given $\mathbf{A}, \mathbf{B} \in \mathrm{GL}_m(q) \times \mathrm{GL}_n(q)$, $\pi_{\mathbf{A},\mathbf{B}}(\mathbf{v})$ is defined as $\mathsf{vector}(\mathbf{A} \cdot \mathsf{matrix}(\mathbf{v}) \cdot \mathbf{B})$. Similarly, $\pi_{\mathbf{A},\mathbf{B}}$ can take a matrix of mn columns as input and outputs the result of applying the operation to each row. Here are some obvious but useful properties of the operator [4] π:

- $\mathbf{G}' = \pi_{\mathbf{A},\mathbf{B}}(\mathbf{G}) \iff \mathbf{G} = \pi_{\mathbf{A}^{-1},\mathbf{B}^{-1}}(\mathbf{G}')$.
- $\pi_{\mathbf{A}' \cdot \mathbf{A}, \mathbf{B} \cdot \mathbf{B}'}(\mathbf{G}) = \pi_{\mathbf{A}',\mathbf{B}'}(\pi_{\mathbf{A},\mathbf{B}}(\mathbf{G}))$.
- $\mathbf{T} \cdot \pi_{\mathbf{A},\mathbf{B}}(\mathbf{G}) = \pi_{\mathbf{A},\mathbf{B}}(\mathbf{T} \cdot \mathbf{G})$.

3 The MEDS Signature Scheme

This section briefly reviews how the MEDS signature scheme [10] was designed. In particular, we explain the underlying hard problem, the underlying Σ-protocol, and how the signature scheme is constructed by applying the Fiat-Shamir transform and various optimizations to the basic Σ-protocol.

3.1 The Matrix Code Equivalence Problem

Definition 1. *Let $\mathcal{C}$ and $\mathcal{D}$ be two $[m \times n, k]$ matrix codes over $\mathbb{F}_q$. We say that $\mathcal{C}$ and $\mathcal{D}$ are equivalent if there exist two matrices $\mathbf{A} \in \mathrm{GL}_m(q)$ and $\mathbf{B} \in \mathrm{GL}_n(q)$ such that $\mathcal{D} = \mathbf{A} \cdot \mathcal{C} \cdot \mathbf{B}$, i.e. for all $\mathbf{C} \in \mathcal{C}$, $\mathbf{A} \cdot \mathbf{C} \cdot \mathbf{B} \in \mathcal{D}$.*

The underlying hard problem in MEDS is the Matrix Code Equivalence (MCE) Problem. The problem is known to be at least as difficult as the Linear Code Equivalence problem (LCE) and as difficult as the Isomorphism of polynomials (IP) problem [7] and the Alternating Trilinear Form Equivalence (ATFE) problem [18,22]. The computational version of it, which is relevant for the MEDS signature construction as well as our optimization is defined as follows.

[4] The operator $\pi_{\mathbf{A},\mathbf{B}}$ can also be represented as the matrix $\mathbf{B}^\top \otimes \mathbf{A}$ where $- \otimes -$ is the Kronecker product.

Public Data
$q, m, n, k \in \mathbb{N}$.
$\mathbf{A}_0 = \mathbf{I}_m \in \mathrm{GL}_m(q)$,
$\mathbf{B}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$.

I. Keygen()
1. $\mathbf{G}_0 \xleftarrow{\$} \mathbb{F}_q^{k \times mn}$.
2. $\mathbf{A}_1, \mathbf{B}_1 \xleftarrow{\$} \mathrm{GL}_m(q) \times \mathrm{GL}_n(q)$.
3. $\mathbf{G}_1, \mathbf{T}_1 \leftarrow \mathsf{SF}^*(\pi_{\mathbf{A}_1,\mathbf{B}_1}(\mathbf{G}_0))$.
   If $\mathbf{G}_1 = \perp$, go to the 2nd step.
4. $\mathsf{sk} \leftarrow (\mathbf{A}_1, \mathbf{B}_1, \mathbf{T}_1)$ and $\mathsf{pk} \leftarrow (\mathbf{G}_0, \mathbf{G}_1)$.
   Return $(\mathsf{sk}, \mathsf{pk})$.

II. Commit(pk)
1. Set $\tilde{\mathbf{A}}, \tilde{\mathbf{B}} \xleftarrow{\$} \mathrm{GL}_m(q) \times \mathrm{GL}_n(q)$.
2. $\tilde{\mathbf{G}} \leftarrow \mathsf{SF}(\pi_{\tilde{\mathbf{A}},\tilde{\mathbf{B}}}(\mathbf{G}_0))$.
   If $\tilde{\mathbf{G}} = \perp$, go to the 1st step.
3. $\mathsf{cmt} \leftarrow \tilde{\mathbf{G}}$. Return cmt.

III. Challenge()
1. $\mathsf{ch} \xleftarrow{\$} \{0, 1\}$. Return ch.

IV. Response(sk, pk, cmt, ch)
1. $\mathsf{rsp} \leftarrow (\tilde{\mathbf{A}} \cdot \mathbf{A}_{\mathsf{ch}}^{-1}, \mathbf{B}_{\mathsf{ch}}^{-1} \cdot \tilde{\mathbf{B}})$.
   Return rsp.

V. Verify(pk, cmt, ch, rsp)
1. Parse rsp into $(\mathbf{A}', \mathbf{B}')$. If $\mathbf{A}' \notin \mathrm{GL}_m(q)$ or $\mathbf{B}' \notin \mathrm{GL}_n(q)$, return "reject".
2. $\mathsf{cmt}' \leftarrow \mathsf{SF}(\pi_{\mathbf{A}',\mathbf{B}'}(\mathbf{G}_{\mathsf{ch}}))$.
   If $\mathsf{cmt}' = \perp$, return "reject".
3. If $\mathsf{cmt}' = \mathsf{cmt}$, return "accept". Otherwise, return "reject".

Fig. 1. MEDS Σ-protocol. Note that $\mathbf{T}_1$ is not used in this protocol actually, but it will be used for the protocols in Figure 2 and Figure 4.

Problem 1 (Matrix Code Equivalence). $\mathsf{MCE}(k, n, m, \mathcal{C}, \mathcal{D})$:
Given: Two k-dimensional matrix codes $\mathcal{C}, \mathcal{D} \subset \mathbb{F}_q^{m \times n}$.
Goal: Find - if any - $\mathbf{A} \in \mathrm{GL}_m(q), \mathbf{B} \in \mathrm{GL}_n(q)$ such that $\mathcal{D} = \mathbf{A} \cdot \mathcal{C} \cdot \mathbf{B}$.

The map $(\mathbf{A}, \mathbf{B}) : \mathbf{C} \mapsto \mathbf{A} \cdot \mathbf{C} \cdot \mathbf{B}$ is called an *isometry* between $\mathcal{C}$ and $\mathcal{D}$, in the sense that it preserves the rank i.e. $\mathrm{Rank}(\mathbf{C}) = \mathrm{Rank}(\mathbf{A} \cdot \mathbf{C} \cdot \mathbf{B})$.

3.2 The MEDS Σ-protocol

The MEDS scheme is based on a three-pass Σ-protocol, of which the security is based on MCE. In matrix notation, the main structure of the protocol is given in Figure 1. The key generation algorithm generates two codes by generating the corresponding generator matrices $\mathbf{G}_0, \mathbf{G}_1$ such that $\mathbf{G}_1 = \mathbf{T}_1 \cdot \pi_{\tilde{\mathbf{A}}_1,\tilde{\mathbf{B}}_1}(\mathbf{G}_0)$. In the commitment algorithm, the prover generates a random isometry $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}})$, applies the isometry to the code generated by $\mathbf{G}_0$, and sets the commitment cmt to (systematic form of a generator matrix of) the resulting code. The prover sets the response rsp to $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}})$ if the challenge ch is equal to 0, or $(\tilde{\mathbf{A}} \cdot \mathbf{A}_1^{-1}, \mathbf{B}_1^{-1} \cdot \tilde{\mathbf{B}})$ if the challenge ch is equal to 1. The verifier parses the response as an isometry $(\mathbf{A}', \mathbf{B}')$, applies the isometry to the code generated by $\mathbf{G}_{\mathsf{ch}} \in \{\mathbf{G}_0, \mathbf{G}_1\}$, and returns "accept" or "reject" depending on whether the resulting code is equal to the code represented by cmt.

3.3 Fiat-Shamir and Common Tricks to Reduce Signature Size

MEDS, as many other signature schemes, is constructed by applying the Fiat-Shamir transform [14] to the underlying Σ-protocol. The signature scheme consists of t rounds of the Σ-protocol. The challenges $\mathsf{ch}^{(0)}, \ldots, \mathsf{ch}^{(t-1)}$ for all rounds are derived from a hash value of the commitments and the message. MEDS makes use of some common techniques to reduce signature size:

1. **Using more than two matrix codes in the public key.** Instead of the matrix code equivalence problem, it is possible to use the **multiple matrix code equivalence problem**, where the attacker is asked to find the isometry between any 2 of s matrix codes. This problem is polynomial-time-equivalent to the MCE problem, therefore there is no need to use it in MEDS in addition to the MCE problem. When applying this optimization, the public key becomes s generator matrices $\mathbf{G}_0, \ldots, \mathbf{G}_{s-1}$, and $(\mathsf{ch}^{(0)}, \ldots, \mathsf{ch}^{(t-1)})$ becomes a random element in $\{0, \ldots, s-1\}^t$ (instead of $\{0,1\}^t$) [13]. This has the benefit of reducing the soundness error to $1/s < 1/2$, and thus it reduces the number of rounds t required in the resulting signature scheme. For simplicity, Figure 1 and all other Σ-protocols in this paper are presented under the assumption $s = 2$. Note that this trick reduces the signature size, at the cost of much larger public keys.

2. **Seeding responses when $\mathsf{ch}^{(i)} = 0$.** The idea is to generate the isometry in the commitment algorithm by expanding a short seed [5]. In this way, signature size can be saved by sending the seed as the response whenever the challenge is 0. Each response still consists of a random isometry when the challenge is nonzero.

3. **Using fixed-weight challenges.** Instead of making $(\mathsf{ch}^{(0)}, \ldots, \mathsf{ch}^{(t-1)})$ a random element in $\{0, \ldots, s-1\}^t$, the idea is to make it a vector of Hamming weight w [5]. w is often chosen to be small compared to t . In this way, the main components of each signature become $t - w$ seeds and w isometries. This helps to reduce the signature size, as an isometry typically takes much more bits to represent than a seed.

4. **Generating seeds from a seed tree.** The previous trick can be refined to obtain even smaller signatures. The idea is to build a binary tree of seeds [5], such that the children of each node can be obtained by hashing the node, together with a salt for the tree to mitigate collision attacks. In this way, the $t - w$ seeds can be compressed into a smaller number of seeds (using a puncturable PRF [16]). The number of required seeds depends on the actual challenges. [8] shows that the number of required seeds is upper bounded by $\mathcal{N}(t, w) := 2^{\lceil \log_2 w \rceil} + w \cdot (\lceil \log_2 t \rceil - \lceil \log_2 w \rceil - 1)$.

Overall, with the tricks above, the signature size in bytes can be calculated as

$$w \cdot \lceil \mathsf{isobits}/8 \rceil + \mathcal{N}(t, w) \cdot \ell_{\text{tree\_seed}} + \ell_{\text{salt}} + \ell_{\text{digest}}. \tag{1}$$

Here, $\mathsf{isobits}$ is the number of bits each response takes when $\mathsf{ch}^{(i)} \neq 0$, which is same as the number of bits every response takes before applying the tricks. Since in Figure 1 each response consists of a matrix in $\mathbb{F}_q^{m \times m}$ and a matrix in $\mathbb{F}_q^{n \times n}$, $\mathsf{isobits}$ is set to be $(m^2+n^2) \cdot \lceil \log_2(q) \rceil$. $\ell_{\text{tree\_seed}}$ is the byte-length of each node in the seed tree. ℓ_{salt} is the byte-length of the salt. ℓ_{digest} is the byte-length of the hash value of the commitments and the message. The values of $\ell_{\text{tree\_seed}}$, ℓ_{salt}, and ℓ_{digest} for the MEDS submission are specified in [10, Table 2].

3.4 Reducing Public-key Size

The multiple public keys optimization discussed in the previous section can reduce the signature size quite substantially, but there is a prize to pay - the public key increases. In order to reduce the impact on the public key of this nice optimization (and to allow for even shorter signatures by reparametrization) a technique for public key compression was introduced in [11]. It basically trades public key size for efficiency, and allows part of the public key to be generated from a public seed, and then be used together with (part of) the secret key to derive the whole secret isometry and the rest of the public key. Typically, the whole isometry can be derived from two seeded codewords in the public key, resulting in reduction of the public key size by $2s(mn-k)$.

4 Optimizing the MEDS Σ-protocol

This section presents our new Σ-protocols for MEDS. Each response in the new Σ-protocols is represented with fewer field elements than those in the original MEDS Σ-protocol (Figure 1) when k is not too large compared to m and n. The new protocols are inspired by the public key compression trick (see Section 3.4).

Before explaining the new protocols, let us introduce some notation and terms from the original MEDS Σ-protocol. We define $\mathcal{C}_i = \{\mathsf{matrix}(\mathbf{v}) \mid \mathbf{v} \in \mathsf{RS}(\mathbf{G}_i)\}$. The code $\mathcal{C}_0$ is considered as the "domain code" for the prover, while $\mathcal{C}_{\mathsf{ch}}$ is considered as the "domain code" for the verifier. In the commitment algorithm, with $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}})$ and $\mathcal{C}_0$, the prover derives its "image code"

$$\tilde{\mathbf{A}} \cdot \mathcal{C}_0 \cdot \tilde{\mathbf{B}} = \left\{\mathsf{matrix}(\mathbf{v}) \mid \mathbf{v} \in \mathsf{RS}(\pi_{\tilde{\mathbf{A}},\tilde{\mathbf{B}}}(\mathbf{G}_0))\right\}$$

and represent it as $\mathsf{cmt} = \mathsf{SF}(\pi_{\tilde{\mathbf{A}},\tilde{\mathbf{B}}}(\mathbf{G}_0))$. Similarly, in the verification algorithm, with $(\mathbf{A}', \mathbf{B}')$ and $\mathcal{C}_{\mathsf{ch}}$, the verifier derives its "image code"

$$\mathbf{A}' \cdot \mathcal{C}_{\mathsf{ch}} \cdot \mathbf{B}' = \left\{\mathsf{matrix}(\mathbf{v}) \mid \mathbf{v} \in \mathsf{RS}(\pi_{\mathbf{A}',\mathbf{B}'}(\mathbf{G}_{\mathsf{ch}}))\right\}$$

and represent it as $\mathsf{cmt}' = \mathsf{SF}(\pi_{\mathbf{A}',\mathbf{B}'}(\mathbf{G}_{\mathsf{ch}}))$. In our new Σ-protocols, each of the prover and verifier still needs to derive the image code from the domain code and an isometry, but the isometry is obtained in a different way.

In our new protocols, codewords in domain codes are represented as their **coordinate vectors**: the coordinate vector of $\mathbf{C} \in \mathcal{C}_i$ (with respect to $\mathbf{G}_i$) is defined as the vector $\mathbf{c} \in \mathbb{F}_q^k$ such that $\mathbf{C} = \mathsf{matrix}(\mathbf{c} \cdot \mathbf{G}_i)$. Such a representation is crucial for reducing the number for field elements each response takes.

For ease of understanding, the reader can simply assume that $k \in \{m, n\}$ in this section, which holds for all parameter sets shown in Section 4.4, and we have $m = n = k$ for all the parameter sets proposed in the MEDS submission [10].

4.1 New Σ-protocol for MEDS with $m = n$

Our first new protocol for MEDS requires that $m = n$. Instead of generating an isometry directly and using it to derive the image code, the prover derives an isometry $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}})$ from a codeword $\tilde{\mathbf{C}} \in \mathrm{GL}_n(q)$ in the domain code and a presumed codeword $\mathbf{D} \in \mathrm{GL}_n(q)$ in the image code such that $\mathbf{D} = \tilde{\mathbf{A}} \cdot \tilde{\mathbf{C}} \cdot \tilde{\mathbf{B}}$. Instead of receiving an isometry directly and using it to derive the image code, the verifier derives an isometry $(\mathbf{A}', \mathbf{B}')$ from a codeword $\mathbf{C}' \in \mathrm{GL}_n(q)$ in the domain code and the same presumed codeword $\mathbf{D} \in \mathrm{GL}_n(q)$ in the image code, such that $\mathbf{D} = \mathbf{A}' \cdot \mathbf{C}' \cdot \mathbf{B}'$. To make sure that there is only 1 possibility for isometry $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}})$ or $(\mathbf{A}', \mathbf{B}')$, $\tilde{\mathbf{A}}$ is generated in the commitment algorithm, and $\mathbf{A}'$ is included in the response, as in Figure 1. Each response then takes only $n^2 + k$ (instead of $2n^2$) field elements: n^2 for $\mathbf{A}'$, and k for the coordinate vector of the codeword $\mathbf{C}'$ in the domain code for the verifier. The codeword $\mathbf{D}$ is considered as public data, so it is not included in rsp.

The new protocol is presented in Figure 2. The public data and the key generation algorithm are mostly same as in Figure 1, except for the public matrices $\mathbf{D} \in \mathrm{GL}_n(q)$ and $\mathbf{T}_0 = \mathbf{I}_k$. In the commitment algorithm, the prover selects a random codeword $\tilde{\mathbf{C}} \in \mathrm{GL}_n(q) \cap \mathcal{C}_0$, generates a random matrix $\tilde{\mathbf{A}} \in \mathrm{GL}_n(q)$, and derives $\tilde{\mathbf{B}}$ as $(\tilde{\mathbf{A}}\tilde{\mathbf{C}})^{-1}\mathbf{D}$. The prover includes $\mathbf{A}' = \tilde{\mathbf{A}} \cdot \mathbf{A}_{\mathsf{ch}}^{-1}$ and the coordinate vector $\tilde{\mathbf{c}} \cdot \mathbf{T}_{\mathsf{ch}}^{-1}$ of $\mathbf{C}' = \mathbf{A}_{\mathsf{ch}} \cdot \tilde{\mathbf{C}} \cdot \mathbf{B}_{\mathsf{ch}} \in \mathrm{GL}_n(q) \cap \mathcal{C}_{\mathsf{ch}}$ in the response. Note that $\mathbf{C}'$ can be viewed as the codeword in $\mathcal{C}_{\mathsf{ch}}$ corresponding to $\tilde{\mathbf{C}}$. In the verification algorithm, the verifier derives $\mathbf{B}'$ as $(\mathbf{A}'\mathbf{C}')^{-1}\mathbf{D}$.

Theorem 1. *The protocol in Figure 2 is a Σ-protocol for the* MCE *relation.*

Proof. In order to show that the protocol is a Σ-protocol we need to prove the properties of completeness, special soundness and honest-verifier zero-knowledge.

Completeness: Given an honestly generated commitment-response pair (cmt_0, rsp_0) for $\mathsf{ch} = 0$ [5] , it is easy to see that $\mathbf{Verify}(\mathsf{pk}, \mathsf{cmt}_0, 0, \mathsf{rsp}_0)$ = "accept" as $\mathbf{C}' = \tilde{\mathbf{C}}$ and $\mathbf{A}' = \tilde{\mathbf{A}}$.

Given an honestly generated commitment-response pair ($\mathsf{cmt}_1, \mathsf{rsp}_1 = (\mathbf{A}', \mathbf{c}')$) for $\mathsf{ch} = 1$, we have

$$\mathbf{C}' = \mathsf{matrix}(\mathbf{c}' \cdot \mathbf{G}_1) = \mathsf{matrix}(\tilde{\mathbf{c}} \cdot \mathbf{T}_1^{-1} \cdot \mathbf{G}_1) = \mathbf{A}_1 \cdot \tilde{\mathbf{C}} \cdot \mathbf{B}_1, \tag{2}$$

and thus $\mathbf{B}' = (\mathbf{A}'\mathbf{C}')^{-1}\mathbf{D} = (\tilde{\mathbf{A}} \cdot \tilde{\mathbf{C}} \cdot \mathbf{B}_1)^{-1}\mathbf{D} = \mathbf{B}_1^{-1} \cdot \tilde{\mathbf{B}}$. Therefore,

$$\mathsf{RS}(\pi_{\mathbf{A}',\mathbf{B}'}(\mathbf{G}_1)) = \mathsf{RS}(\pi_{\tilde{\mathbf{A}}\cdot\mathbf{A}_1^{-1},\mathbf{B}_1^{-1}\cdot\tilde{\mathbf{B}}}(\mathbf{G}_1)) = \mathsf{RS}(\pi_{\tilde{\mathbf{A}},\tilde{\mathbf{B}}}(\mathbf{G}_0)), \tag{3}$$

which implies that $\mathsf{SF}(\pi_{\mathbf{A}',\mathbf{B}'}(\mathbf{G}_1)) = \mathsf{SF}(\pi_{\tilde{\mathbf{A}},\tilde{\mathbf{B}}}(\mathbf{G}_0)) = \mathsf{cmt}_1$.

[5] "Honestly generated" means that the pair is obtained by running $\mathsf{cmt}_0 \leftarrow \mathbf{Commit}(\mathsf{pk})$ and then $\mathsf{rsp}_0 \leftarrow \mathbf{Response}(\mathsf{sk}, \mathsf{pk}, \mathsf{cmt}_0, \mathsf{ch})$.

Public Data

$q, m, n, k \in \mathbb{N}$, such that $m = n$.
$\mathbf{A}_0 = \mathbf{B}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$. $\mathbf{T}_0 = \mathbf{I}_k \in \mathrm{GL}_k(q)$.
$\mathbf{D} \in \mathrm{GL}_n(q)$.

II. Commit(pk)

1. Set $\tilde{\mathbf{c}} \xleftarrow{\$} \mathbb{F}_q^k$. $\tilde{\mathbf{C}} \leftarrow \mathrm{matrix}(\tilde{\mathbf{c}} \cdot \mathbf{G}_0)$. If $\tilde{\mathbf{C}} \notin \mathrm{GL}_n(q)$, repeat this step.
2. Set $\tilde{\mathbf{A}} \xleftarrow{\$} \mathrm{GL}_n(q)$. $\tilde{\mathbf{B}} \leftarrow (\tilde{\mathbf{A}}\tilde{\mathbf{C}})^{-1}\mathbf{D}$.
3. $\tilde{\mathbf{G}} \leftarrow \mathsf{SF}(\pi_{\tilde{\mathbf{A}},\tilde{\mathbf{B}}}(\mathbf{G}_0))$. If $\tilde{\mathbf{G}} = \perp$, go to the 1st step.
4. $\mathsf{cmt} \leftarrow \tilde{\mathbf{G}}$. Return cmt.

III. Challenge()

1. $\mathsf{ch} \xleftarrow{\$} \{0, 1\}$. Return ch.

IV. Response($\mathsf{sk}, \mathsf{pk}, \mathsf{cmt}, \mathsf{ch}$)

1. $\mathsf{rsp} \leftarrow (\tilde{\mathbf{A}} \cdot \mathbf{A}_{\mathsf{ch}}^{-1}, \tilde{\mathbf{c}} \cdot \mathbf{T}_{\mathsf{ch}}^{-1})$. Return rsp.

V. Verify($\mathsf{pk}, \mathsf{cmt}, \mathsf{ch}, \mathsf{rsp}$)

1. Parse rsp into $(\mathbf{A}', \mathbf{c}')$. If $\mathbf{A}' \notin \mathrm{GL}_n(q)$, return "reject".
2. $\mathbf{C}' \leftarrow \mathrm{matrix}(\mathbf{c}' \cdot \mathbf{G}_{\mathsf{ch}})$. If $\mathbf{C}' \notin \mathrm{GL}_n(q)$, return "reject".
3. $\mathbf{B}' \leftarrow (\mathbf{A}'\mathbf{C}')^{-1}\mathbf{D}$.
4. $\mathsf{cmt}' \leftarrow \mathsf{SF}(\pi_{\mathbf{A}',\mathbf{B}'}(\mathbf{G}_{\mathsf{ch}}))$. If $\mathsf{cmt}' = \perp$, return "reject".
5. If $\mathsf{cmt} = \mathsf{cmt}'$, return "accept". Otherwise, return "reject".

Fig. 2. New MEDS Sigma Protocol for $m = n$. Keygen as in Figure 1.

Zero-knowledge: To simulate a commitment-response pair for $\mathsf{ch} = 0$, apparently the simulator can simply run $\hat{\mathsf{cmt}}_0 \leftarrow \mathbf{Commit}(\mathsf{pk})$ and also provide $\hat{\mathsf{rsp}}_0 \leftarrow \mathbf{Response}(\mathsf{sk}^*, \mathsf{pk}, \hat{\mathsf{cmt}}_0, 0)$, where sk^* is a dummy argument that is not actually used.

Let $\mathbf{Commit}^{(\mathbf{G}_1)}$ be the modified version of $\mathbf{Commit}$ where $\mathbf{G}_0$ is replaced by $\mathbf{G}_1$. We claim that for $\mathsf{ch} = 1$, the simulator can simulate a commitment-response pair by running $\hat{\mathsf{cmt}}_1 \leftarrow \mathbf{Commit}^{(\mathbf{G}_1)}(\mathsf{pk})$ and correspondingly $\hat{\mathsf{rsp}}_1 = (\hat{\mathbf{A}}, \hat{\mathbf{c}}) \leftarrow \mathbf{Response}(\mathsf{sk}^*, \mathsf{pk}, \hat{\mathsf{cmt}}_1, 0)$. In other words, we claim that an honestly generated pair $(\mathsf{cmt}_1, \mathsf{rsp}_1 = (\tilde{\mathbf{A}} \cdot \mathbf{A}_1^{-1}, \tilde{\mathbf{c}} \cdot \mathbf{T}_1^{-1}))$ for $\mathsf{ch} = 1$ has exactly the same distribution as $(\hat{\mathsf{cmt}}_1, \hat{\mathsf{rsp}}_1)$.

To see that the claim is true, first consider a modified version of the protocol, where each SF is replaced by RREF (replacement also happens in $\mathbf{Commit}^{(\mathbf{G}_1)}$). With procedures in this modified protocol, $\hat{\mathbf{A}}$ will be a uniform random element in $\mathrm{GL}_n(q)$, and $\hat{\mathbf{c}}$ will be the coordinate vector of a uniform random element in $\mathcal{C}_1 \cap \mathrm{GL}_n(q)$. $(\tilde{\mathbf{A}} \cdot \mathbf{A}_1^{-1}, \tilde{\mathbf{c}} \cdot \mathbf{T}_1^{-1})$ has the same distribution as $(\hat{\mathbf{A}}, \hat{\mathbf{c}})$: $\tilde{\mathbf{A}}$ is uniform random in $\mathrm{GL}_n(q)$, and *Equation* (2) shows that $\tilde{\mathbf{c}} \cdot \mathbf{T}_1^{-1}$ has the same distribution as $\hat{\mathbf{c}}$.

Then, the proof of completeness shows that there is a deterministic procedure that can be used to derive cmt_1 from $(\mathsf{rsp}_1, \mathbf{G}_1, \mathbf{D})$ and to derive $\hat{\mathsf{cmt}}_1$ from $(\hat{\mathsf{rsp}}_1, \mathbf{G}_1, \mathbf{D})$, so $(\mathsf{rsp}_1, \mathsf{cmt}_1)$ has the same distribution as $(\hat{\mathsf{rsp}}_1, \hat{\mathsf{cmt}}_1)$. Finally, changing RREF back to SF simply means adding the same constraint to cmt_1 and $\hat{\mathsf{cmt}}_1$, so $(\mathsf{rsp}_1, \mathsf{cmt}_1)$ still has the same distribution as $(\hat{\mathsf{rsp}}_1, \hat{\mathsf{cmt}}_1)$ when the original protocol is considered.

Input: $\mathbf{C}_0, \mathbf{C}_1, \mathbf{D}_0, \mathbf{D}_1 \in \mathbb{F}_q^{m \times n}$.
Output: $(\mathbf{A}, \mathbf{B}) \in \mathrm{GL}_m(q) \times \mathrm{GL}_n(q)$ such that $\mathbf{AC}_0\mathbf{B} = \mathbf{D}_0$ and $\mathbf{AC}_1\mathbf{B} = \mathbf{D}_1$, or $\perp$.

1. Build a linear system of $2nm$ equations and $m^2 + n^2$ variables by considering
$$\mathbf{AC}_0 = \mathbf{D}_0\mathbf{B}^{-1}, \mathbf{AC}_1 = \mathbf{D}_1\mathbf{B}^{-1},$$
where entries in $\mathbf{A}$ and $\mathbf{B}^{-1}$ are considered as variables.
2. If the solution space of the linear system does not have dimension 1, return $\perp$.
3. Pick any nontrivial solution in the solution space to obtain $(\mathbf{A}, \mathbf{B}^{-1}) \in \mathbb{F}_q^{m \times m} \times \mathbb{F}_q^{n \times n}$. If $\mathbf{A} \notin \mathrm{GL}_m(\mathbb{F}_q)$ or $\mathbf{B}^{-1} \notin \mathrm{GL}_n(\mathbb{F}_q)$, return $\perp$. Otherwise, return $(\mathbf{A}, \mathbf{B})$.

Fig. 3. The function **Solve**.

Special soundness: Given $\mathsf{cmt}, \mathsf{rsp}_0$, rsp_1 such that $\mathbf{Verify}(\mathsf{pk}, \mathsf{cmt}, 0, \mathsf{rsp}_0) =$ "accept" $= \mathbf{Verify}(\mathsf{pk}, \mathsf{cmt}, 1, \mathsf{rsp}_1)$, there exists an efficient algorithm that outputs $\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \mathbf{A}', \mathbf{B}'$, such that

$$\mathsf{SF}(\pi_{\tilde{\mathbf{A}}, \tilde{\mathbf{B}}}(\mathbf{G}_0)) = \mathsf{cmt} = \mathsf{SF}(\pi_{\mathbf{A}', \mathbf{B}'}(\mathbf{G}_1)).$$

Then, $((\mathbf{A}')^{-1} \cdot \tilde{\mathbf{A}}, \tilde{\mathbf{B}} \cdot (\mathbf{B}')^{-1})$ yields a solution to the MCE instance $(\mathbf{G}_0, \mathbf{G}_1)$. □

One might expect that the protocol still works:

- if the prover simply picks an arbitrary $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}})$ such that $\tilde{\mathbf{A}} \cdot \tilde{\mathbf{C}} \cdot \tilde{\mathbf{B}} = \mathbf{D}$, and
- if the verifier simply picks an arbitrary $(\mathbf{A}', \mathbf{B}')$ such that $\mathbf{A}' \cdot \mathbf{C}' \cdot \mathbf{B}' = \mathbf{D}$.

In this way, rsp does not need to include $\mathbf{A}'$. However, now it is no longer guaranteed that $(\mathbf{A}', \mathbf{B}') = (\tilde{\mathbf{A}} \cdot \mathbf{A}_{\mathsf{ch}}^{-1}, \mathbf{B}_{\mathsf{ch}}^{-1} \cdot \tilde{\mathbf{B}})$. Consequently, it is very likely that $\mathsf{SF}(\pi_{\tilde{\mathbf{A}}, \tilde{\mathbf{B}}}(\mathbf{G}_0)) \neq \mathsf{SF}(\pi_{\mathbf{A}', \mathbf{B}'}(\mathbf{G}_{\mathsf{ch}}))$, which means completeness is broken.

Variants of the protocol. There are some natural variants of the protocol. For example, instead of having $\mathbf{D}$ as public data, $\mathbf{D}$ can be considered as a part of the public key. As another example, $\mathbf{D}$ can be chosen as a random invertible matrix and included in the response. The proof for Theorem 1 also applies to these variants. We note that in the second variant, to save signature size, the signer can send a short seed from which $\mathbf{D}$'s for all rounds can be derived in a pseudo-random fashion.

4.2 New Σ-protocol for MEDS with $|m - n| \leq 1$

The second new Σ-protocol for MEDS requires that $|m - n| \leq 1$. The main idea of the new protocol is that each of the prover and verifier derives an isometry that maps a pair of codewords in the domain code to a pair of presumed codewords

Public Data

$q, m, n, k \in \mathbb{N}$, such that $|m - n| \leq 1$.
$\mathbf{A}_0 = \mathbf{I}_m \in \mathrm{GL}_m(q)$.
$\mathbf{B}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$.
$\mathbf{T}_0 = \mathbf{I}_k \in \mathrm{GL}_k(q)$.
Full-rank, linearly independent $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{F}_q^{m \times n}$.

II. Commit(pk)

1. Set $(\tilde{\mathbf{c}}_0, \tilde{\mathbf{c}}_1) \xleftarrow{\$} \mathbb{F}_q^k \times \mathbb{F}_q^k$. If $\tilde{\mathbf{c}}_0$ and $\tilde{\mathbf{c}}_1$ are linearly dependent, repeat this step.
2. For $i \in \{0, 1\}$, $\tilde{\mathbf{C}}_i \leftarrow \mathrm{matrix}(\tilde{\mathbf{c}}_i \cdot \mathbf{G}_0)$. If $\tilde{\mathbf{C}}_0$ or $\tilde{\mathbf{C}}_1$ is not full rank, go to the 1st step.
3. $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}}) \leftarrow \mathrm{Solve}(\tilde{\mathbf{C}}_0, \tilde{\mathbf{C}}_1, \mathbf{D}_0, \mathbf{D}_1)$. If $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}}) = \perp$, go to the 1st step.
4. $\tilde{\mathbf{G}} \leftarrow \mathsf{SF}(\pi_{\tilde{\mathbf{A}}, \tilde{\mathbf{B}}}(\mathbf{G}_0))$. If $\tilde{\mathbf{G}} = \perp$, go to the 1st step.
5. $\mathsf{cmt} \leftarrow \tilde{\mathbf{G}}$. Return cmt.

IV. Response(sk, pk, cmt, ch)

1. $\mathsf{rsp} \leftarrow (\tilde{\mathbf{c}}_0 \cdot \mathbf{T}_{\mathsf{ch}}^{-1}, \tilde{\mathbf{c}}_1 \cdot \mathbf{T}_{\mathsf{ch}}^{-1})$. Return rsp.

III. Challenge()

1. $\mathsf{ch} \xleftarrow{\$} \{0, 1\}$. Return ch.

V. Verify(pk, cmt, ch, rsp)

1. Parse rsp into $(\mathbf{c}'_0, \mathbf{c}'_1)$. If $\mathbf{c}'_0$ and $\mathbf{c}'_1$ are linearly dependent, return "reject".
2. For $i \in \{0, 1\}$, $\mathbf{C}'_{\mathbf{i}} \leftarrow \mathrm{matrix}(\mathbf{c}'_i \cdot \mathbf{G}_{\mathsf{ch}})$. If $\mathbf{C}'_0$ or $\mathbf{C}'_1$ is not full rank, return "reject".
3. $(\mathbf{A}', \mathbf{B}') \leftarrow \mathrm{Solve}(\mathbf{C}'_0, \mathbf{C}'_1, \mathbf{D}_0, \mathbf{D}_1)$. If $(\mathbf{A}', \mathbf{B}') = \perp$, return "reject".
4. $\mathsf{cmt}' \leftarrow \mathsf{SF}(\pi_{\mathbf{A}', \mathbf{B}'}(\mathbf{G}_{\mathsf{ch}}))$. If $\mathsf{cmt}' = \perp$, return "reject".
5. If $\mathsf{cmt} = \mathsf{cmt}'$, return "accept". Otherwise, return "reject".

Fig. 4. New MEDS Sigma Protocol for $|m - n| \leq 1$. Keygen as in Figure 1.

in the image code. The prover generated the codewords in the domain code. For the verifier, the codewords in the domain code are included in the response. Since the codewords in the response are represented as coordinate vectors, the response takes only $2k$ field elements to represent. As in the protocol shown in the previous subsection, the presumed codewords are considered as public data, so they are not included in the response.

The prover and verifier make use of the procedure **Solve** (c.f. Figure 3) to derive an isometry mapping $\mathbf{C}_0, \mathbf{C}_1 \in \mathbb{F}_q^{m \times n}$ in the domain code to $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{F}_q^{m \times n}$ in the image code. The procedure computes an isometry by solving the linear system of $2mn$ equations and $m^2 + n^2$ equations formed by

$$\mathbf{A}\mathbf{C}_0 = \mathbf{D}_0\mathbf{B}^{-1}, \mathbf{A}\mathbf{C}_1 = \mathbf{D}_1\mathbf{B}^{-1},$$

where entries in $\mathbf{A}$ and $\mathbf{B}^{-1}$ are considered as variables. Note that **Solve** is allowed to return any $(\mathbf{A}, \mathbf{B})$ as long as $(\mathbf{A}, \mathbf{B}^{-1}) \in \mathrm{GL}_m(q) \times \mathrm{GL}_n(q)$ is in the solution space, under the condition that the solution space has dimension exactly 1. This makes sense: Under the condition, the set of all valid solutions must be of the form $\{(\alpha\mathbf{A}, \alpha\mathbf{B}^{-1}) \mid \alpha \in \mathbb{F}_q^*\}$, and $\pi_{\alpha\mathbf{A}, \alpha^{-1}\mathbf{B}}(\mathbf{G})$ is independent of the choice of α.

In the new protocol shown in Figure 4, the public data and the key generation algorithm is mostly same as in Figure 1, except for the public matrices $\mathbf{T}_0 = \mathbf{I}_k$

and $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{F}_q^{m \times n}$. In the commitment algorithm, the prover generates two full-rank, linearly independent codewords $\tilde{\mathbf{C}}_0, \tilde{\mathbf{C}}_1 \in \mathcal{C}_0$ and uses **Solve** to find an isometry $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}})$ that maps $\tilde{\mathbf{C}}_0, \tilde{\mathbf{C}}_1$ to $\mathbf{D}_0, \mathbf{D}_1$. The response $(\tilde{\mathbf{c}}_0 \cdot \mathbf{T}_{\mathsf{ch}}^{-1}, \tilde{\mathbf{c}}_1 \cdot \mathbf{T}_{\mathsf{ch}}^{-1})$ is essentially the coordinate vectors (w.r.t. $\mathbf{G}_{\mathsf{ch}}$) of the two codewords in $\mathcal{C}_{\mathsf{ch}}$ corresponding to $\tilde{\mathbf{C}}_0$ and $\tilde{\mathbf{C}}_1$. In the verification algorithm, the verifier derives two codewords $\mathbf{C}'_0, \mathbf{C}'_1 \in \mathcal{C}_{\mathsf{ch}}$ from the coordinate vectors in the response, and uses **Solve** to find an isometry $(\mathbf{A}', \mathbf{B}')$ that maps $\mathbf{C}'_0, \mathbf{C}'_1$ to $\mathbf{D}_0, \mathbf{D}_1$.

As in the previous section, $\mathbf{D}_i$'s can also be considered as a part of the public key or chosen as random matrices. The proof for the following theorem also applies to these variants.

Theorem 2. *The protocol in Figure 4 is a Σ-protocol for the* MCE *relation.*

Proof. We show completeness, zero knowledge, and special soundness:

Completeness. Given an honestly generated commitment-response pair (cmt_0, rsp_0) for $\mathsf{ch} = 0$, it is easy to see that $\mathbf{Verify}(\mathsf{pk}, \mathsf{cmt}_0, 0, \mathsf{rsp}_0) =$ "accept" as $(\mathbf{A}', \mathbf{B}') = (\alpha\tilde{\mathbf{A}}, \alpha^{-1}\tilde{\mathbf{B}})$ for some $\alpha \in \mathbb{F}_q^*$.

Given an honestly generated commitment-response pair $(\mathsf{cmt}_1, \mathsf{rsp}_1 = (\mathbf{c}'_0, \mathbf{c}'_1))$ for $\mathsf{ch} = 1$, we have $\mathbf{C}'_i = \mathbf{A}_1 \cdot \tilde{\mathbf{C}}_i \cdot \mathbf{B}_1$ for $i \in \{0, 1\}$ according to the completeness proof of Theorem 1. This implies that

$$\mathbf{A} \cdot \mathbf{C}'_i \cdot \mathbf{B} = \mathbf{D}_i, \text{ for } i \in \{0,1\} \implies \mathbf{A}\mathbf{A}_1 \cdot \tilde{\mathbf{C}}_i \cdot \mathbf{B}_1\mathbf{B} = \mathbf{D}_i, \text{ for } i \in \{0,1\}$$

and

$$\mathbf{A} \cdot \tilde{\mathbf{C}}_i \cdot \mathbf{B} = \mathbf{D}_i, \text{ for } i \in \{0,1\} \implies \mathbf{A}\mathbf{A}_1^{-1} \cdot \mathbf{C}'_i \cdot \mathbf{B}_1^{-1}\mathbf{B} = \mathbf{D}_i, \text{ for } i \in \{0,1\}.$$

In other words, there is a bijective map between the set of $(\mathbf{A}, \mathbf{B})$'s mapping $\tilde{\mathbf{C}}_i$'s to $\mathbf{D}_i$'s and the set of $(\mathbf{A}, \mathbf{B})$'s mapping $\mathbf{C}'_i$'s to $\mathbf{D}_i$'s, and thus the two sets have the same cardinality. Therefore, the solution space for $\mathbf{A}', (\mathbf{B}')^{-1}$ (in the verification algorithm) must be of dimension 1, and $(\mathbf{A}', \mathbf{B}') = (\alpha\tilde{\mathbf{A}}\mathbf{A}_1^{-1}, \alpha^{-1}\mathbf{B}_1^{-1}\tilde{\mathbf{B}})$ for some $\alpha \in \mathbb{F}_q^*$. We conclude that Equation (3) holds and thus $\mathsf{SF}(\pi_{\mathbf{A}',\mathbf{B}'}(\mathbf{G}_1)) = \mathsf{SF}(\pi_{\tilde{\mathbf{A}},\tilde{\mathbf{B}}}(\mathbf{G}_0)) = \mathsf{cmt}_1$.

Zero knowledge. The $\mathsf{ch} = 0$ case is trivial as in the proof for zero knowledge for Theorem 1. We claim that for $\mathsf{ch} = 1$, the simulator can simulate a commitment-response pair by running $\hat{\mathsf{cmt}}_1 \leftarrow \mathbf{Commit}^{(\mathbf{G}_1)}(\mathsf{pk})$, and $\hat{\mathsf{rsp}}_1 = (\hat{\mathbf{c}}_0, \hat{\mathbf{c}}_1) \leftarrow \mathbf{Response}(\mathsf{sk}^*, \mathsf{pk}, \hat{\mathsf{cmt}}_1, 0)$. That is, we claim that honestly generated $(\mathsf{cmt}_1, \mathsf{rsp}_1 = (\tilde{\mathbf{c}}_0 \cdot \mathbf{T}_1^{-1}, \tilde{\mathbf{c}}_1 \cdot \mathbf{T}_1^{-1}))$ for $\mathsf{ch} = 1$ has the same distribution as $(\hat{\mathsf{cmt}}_1, \hat{\mathsf{rsp}}_1)$.

We follow the proof for zero knowledge for Theorem 1 to first consider the modified protocol where SF is replaced by RREF. With the modified procedures, $(\hat{\mathbf{c}}_0, \hat{\mathbf{c}}_1)$ is going to be coordinate vectors (w.r.t. $\mathbf{G}_1$) of 2 uniform random codewords $\hat{\mathbf{C}}_0, \hat{\mathbf{C}}_1 \in \mathcal{C}_1$ such that

- $\hat{\mathbf{C}}_0, \hat{\mathbf{C}}_1$ are linearly independent and full-rank and
- $\mathbf{Solve}(\hat{\mathbf{C}}_0, \hat{\mathbf{C}}_1, \mathbf{D}_0, \mathbf{D}_1) \neq \perp$.

The proof for completeness shows that $(\tilde{\mathbf{c}}_0 \cdot \mathbf{T}_1^{-1}, \tilde{\mathbf{c}}_1 \cdot \mathbf{T}_1^{-1})$ has the same distribution as $(\hat{\mathbf{c}}_0, \hat{\mathbf{c}}_1)$. The remainder of the proof is similar to the proof for zero knowledge for Theorem 1.

Special soundness. The proof is essentially the same as the one for Theorem 1, so we do not repeat it here. □

Why is the dimension of the solution space restricted to 1 in **Solve***?* When the dimension is 0, the only solution of the linear system will lead to $(\mathbf{A}, \mathbf{B}^{-1}) = (0, 0)$, which is not desired. When the dimension is larger than 1, it is no longer guaranteed that $(\mathbf{A}', \mathbf{B}') = (\alpha \tilde{\mathbf{A}} \mathbf{A}_1^{-1}, \alpha^{-1} \mathbf{B}_1^{-1} \tilde{\mathbf{B}})$, so completeness is broken. This also explains why we require that $|m - n| \leq 1$: When $|m - n| > 1$, we have $2mn + 1 < m^2 + n^2$, so the dimension of the solution space will be larger than 1.

Success probability of **Solve**. The linear systems in **Solve** are highly structured and should not be considered as random linear systems. Nevertheless, for most $\mathbf{D}_i$'s, experiment results show that the probability **Solve** returns an isometry is close to the probability that a random matrix in $\mathbb{F}_q^{2mn \times (m^2+n^2)}$ has a kernel space of dimension 1. This means that when $m = n$, **Solve** returns $\perp$ with a high probability, so the signing algorithm is expected to be inefficient. In the next subsection, we give some arguments regarding the success probability (and complexity) of **Solve** for a specific choice of $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{F}_q^{m \times n}$ with $n = m + 1$.

4.3 A Specific Choice for $\mathbf{D_0}, \mathbf{D_1}$

For the $|m-n| \leq 1$ protocol (c.f. Figure 4) introduced in the previous subsection, the linear systems in **Solve** are of the form $\mathbf{Lx} = 0$, where $\mathbf{L} \in \mathbb{F}_q^{2mn \times (m^2+n^2)}$. In general, one would expect that it takes $\mathrm{O}(n^6)$ field operations to solve such a linear system. However, we found that, due to setting $n = m + 1$ and by using $\mathbf{D}_0 = (\mathbf{I}_m \mid 0) \in \mathbb{F}_q^{m \times n}$, $\mathbf{D}_1 = (0 \mid \mathbf{I}_m) \in \mathbb{F}_q^{m \times n}$, we can obtain a much more sparse and structured linear system such that the cost can be reduced to only $\mathrm{O}(n^3)$ field operations.

With such a choice for $\mathbf{D}_i$'s, the matrix $\mathbf{L}$ is going to be in the shape illustrated in Figure 5a. After various row operations are applied, the matrix is transformed into the form illustrated by Figure 5f, which is almost in row echelon form. Then, computing reduced row echelon form of $\mathbf{L}$ boils down to computing reduced row echelon form of $\mathbf{M} \in \mathbb{F}_q^{(m-1) \times m}$ plus backward substitutions. Note that checking whether $\mathbf{L}$ has a dimension-1 kernel space also boils down to checking whether $\mathbf{M}$ has a dimension-1 kernel space. Below we explain why $\mathbf{L}$ is of the shape of Figure 5a and how the matrix in Figure 5f is obtained by applying a sequence of row operations.

Let $a_{i,j}$ be the variable for $\mathbf{A}_{i,j}$, and let $b_{i,j}$ be the variable for $(\mathbf{B}^{-1})_{i,j}$. Consider $\mathbf{D}_0\mathbf{B}^{-1} - \mathbf{A}\mathbf{C}_0 = 0$. We have $(\mathbf{A}\mathbf{C}_0)_{i,j} = \sum_k \mathbf{A}_{i,k}(\mathbf{C}_0)_{k,j}$, and $\mathbf{D}_0\mathbf{B}^{-1}$ is simply the first m rows of $\mathbf{B}^{-1}$. This leads to equations $b_{i,j} - \sum_{k=0}^{m-1} (\mathbf{C}_0)_{k,j} a_{i,k}$, for $i = 0, \ldots, n-2$ and $j = 0, \ldots, n-1$. This explains the first mn rows of

Figure 5a: for any (i, j), the equation is represented in row $in + j$ of $\mathbf{L}$, while the first n^2 columns of $\mathbf{L}$ correspond to

$$b_{0,0}, \ldots, b_{0,n-1}, b_{1,0}, \ldots, b_{1,n-1}, \ldots, b_{n-1,0}, \ldots, b_{n-1,n-1},$$

and the last m^2 columns of $\mathbf{L}$ correspond to

$$a_{0,0}, \ldots, a_{0,m-1}, a_{1,0}, \ldots, a_{1,m-1}, \ldots, a_{m-1,0}, \ldots, a_{m-1,m-1},$$

Following the discussion above, it is easy to see that the last mn rows of Figure 5a are for $\mathbf{D}_1\mathbf{B}^{-1} - \mathbf{A}\mathbf{C}_1 = 0$.

Figure 5f is derived from Figure 5a by carrying out the steps below.

- Figure 5b is obtained by rearranging rows in Figure 5a. This step takes 0 field operations.

- Figure 5c is obtained by eliminating the $\mathbf{I}_n$'s in the last $(m-1)n$ rows of Figure 5b. This step takes 0 field operations as we only need to replace the $\mathbf{I}_n$'s by 0 and replace some zero submatrices by $\mathbf{C}_0^\mathsf{T}$.

- Figure 5d is obtained by applying Gaussian eliminations to the $(-\mathbf{C}_1^\mathsf{T} \mid \mathbf{C}_0^\mathsf{T})$ blocks in Figure 5c. Note that we can always reduce $(-\mathbf{C}_1^\mathsf{T} \mid \mathbf{C}_0^\mathsf{T})$ to the form

  $$\begin{bmatrix} \mathbf{I}_m & \mathbf{N} \\ 0 & \mathbf{v} \end{bmatrix}$$

  where $\mathbf{v} \in \mathbb{F}_q^m$, as $\mathbf{C}_1$ is full rank. This step takes $O(n^2 m) = O(n^3)$ field operations.

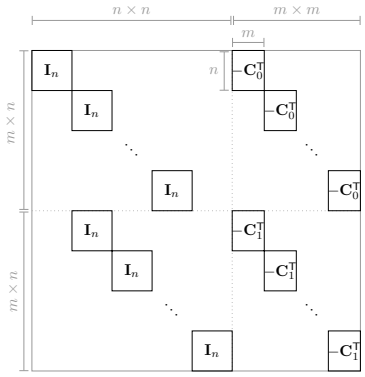
- Figure 5e is obtained by rearranging rows in Figure 5d. This step takes 0 field operations.

- Figure 5f is obtained by eliminating elements in the last $m-1$ rows in Figure 5e. It turns out that the rows of $\mathbf{M}$ are determined by $\mathbf{v}$ and $\mathbf{N}$ as we have:

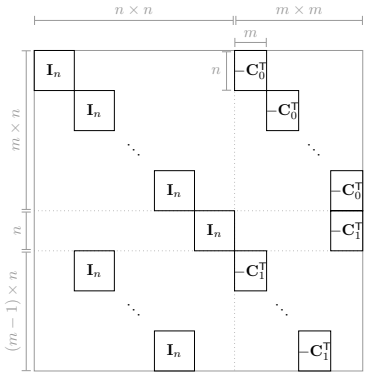
  $$\mathbf{M} = \begin{bmatrix} \mathbf{v} \cdot (-\mathbf{N})^0 \\ \mathbf{v} \cdot (-\mathbf{N})^1 \\ \mathbf{v} \cdot (-\mathbf{N})^2 \\ \vdots \\ \mathbf{v} \cdot (-\mathbf{N})^{m-2} \end{bmatrix} \in \mathbb{F}_q^{(m-1)\times m}.$$

  Therefore, this step takes $O(n^3)$ field operations, if each row is computed as the product of the previous row and $-\mathbf{N}$.

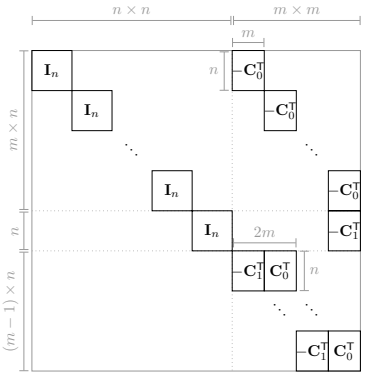
Computing reduced row echelon form of $\mathbf{M}$ takes $O(n^3)$ field operations. Backward substitutions also take $O(n^3)$ field operations. Checking whether $\mathbf{A}$ and $\mathbf{B}^{-1}$ are invertible again takes $O(n^3)$ field operations. Therefore, we conclude that **Solve** takes $O(n^3)$ field operations in total. For comparison, each of the $\pi_{\tilde{\mathbf{A}},\tilde{\mathbf{B}}}$, $\pi_{\mathbf{A}',\mathbf{B}'}$, and SF operations in the commitment and verification algorithms takes $O(n^4)$ field operations, under the asssumption that $k = O(n)$.



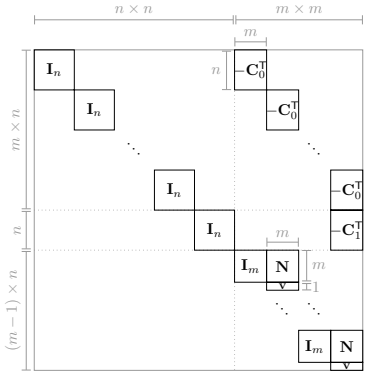
(a) Initial sparse system matrix.



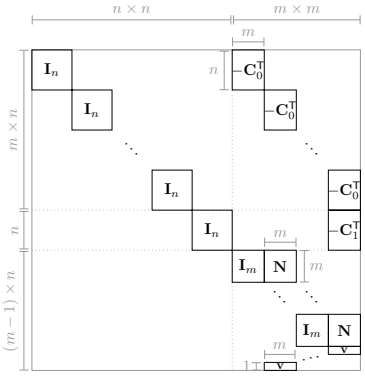
(b) After moving the last block of m rows upwards.



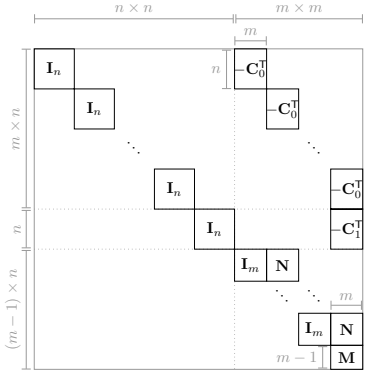
(c) After eliminating the left bottom quadrant.



(d) After Gaussian elimination of the subsystem $(-\mathbf{C}_1^\mathsf{T}|\mathbf{C}_0^\mathsf{T})$.



(e) After moving the last row of each row block downwards.



(f) After elimination of the last $m-1$ rows, which gives subsystem $\mathbf{M}$.

Fig. 5. Process of solving the block diagonal system.

Success Probability. According to the discussion above, the matrix $\mathbf{L}$ is full rank if $\mathbf{M}$ is full rank. Whether $\mathbf{M}$ is full rank depends on whether the dimension of the Krylov space generated by $\mathbf{N}$ and $\mathbf{v}$ is at least $m-1$. According to [15], for a random matrix in $\mathbb{F}_q^{m\times m}$, the $m\to\infty$ limit of the probability that its minimal polynomial is the same as its characteristic polynomial (such a matrix is called a nonderogatory matrix), is $(1-\frac{1}{q^5})/(1+\frac{1}{q^3})$. According to [9, Theorem 9], the probability that the Krylov space generated by a nonderogatory matrix in $\mathbb{F}_q^{m\times m}$ and a random vector $\mathbf{v}\in\mathbb{F}_q^m$ is of dimension m, is lower bounded by $\frac{0.218}{1+\log_q m}$. For parameter sets with $n=m+1$ shown in the next subsection, q is always much larger than m, and experiment results show that the success probability of **Solve** is much closer to 1 than to 0.218, presumably because the dimension of the Krylov space is only required to be at least $m-1$.

Constant time implementation. For the signing operation, system solving must be computed in constant time to avoid leakage of secret information via timing variations. Computing $\mathbf{N}$ and v in constant time from $(-\mathbf{C}_1^\mathsf{T} \mid \mathbf{C}_0^\mathsf{T})$ is very similar to computing the systematic form in constant time — the same row operations as for systemization are simply only performed for the first m rows (while including the last row in pivoting and elimination).

Solving the subsystem $\mathbf{M}$ requires more effort since we need to compute not a systematic form but a reduced row echelon form from $\mathbf{M}$. This can be an expensive computation when performed in constant time, since all following columns need to be conditionally swapped with with pivoting column to make sure to find a pivot. Nevertheless, since we have the condition that the overall system and hence $\mathbf{M}$ must have a solution space of dimension one, it is sufficient to perform a single conditional swap with the last column during each pivoting operation. We count how often a column swap actually is necessary: If more than one swap needs to be performed, $\mathbf{M}$ has more than one solution and system solving is aborted with an error code. If $\mathbf{M}$ can be solved successfully, to obtain a complete constant time operation, we need to perform conditional column swaps on the remaining system matrix for back-substitution (logically, since the system matrix never is generated completely) and finally on the solution vector.

4.4 Results

Table 1 shows the impact of the $|m-n|\leq 1$ Σ-protocol from Section 4.2 on the signature sizes, public key sizes, and signing time, when $\mathbf{D}_0$ and $\mathbf{D}_1$ are chosen in the way as discussed in Section 4.3 (which means we have $n-m=1$). The verification time is similar to signing time without further verification-specific optimization and hence not explicitly shown in the table. We integrated our optimization as proof-of-concept into the public source code of the MEDS reference implementation[6]. For a fair comparison, as the reference implementation, our modifications are constant time. The cycle counts were obtained on a AMD Ryzen 7 PRO 5850U.

[6] `https://github.com/MEDSpqc/meds`

For each parameter set, the first row ("spec [10]") shows the performance of the original MEDS reference implementation without our optimization. Also the second row ("spec n++") is without our optimization, but n is changed to $m+1$ (i.e., incremented by one) to identify what impact on performance is due to the parameter change and what impact is due to the additional computation for the signature size optimization. The third row ("sig opt") shows the performance including our $|m - n| \leq 1$ signature size optimization from Section 4.2. The signature size for the "sig opt" cases can be computed using Equation (1) and by setting $\mathsf{isobits}$ to $2k\lceil \log_2(q) \rceil$.

In all six parameter set cases, the increase in computing time for signing when using the signature optimization compared to the original parameter set is less than 50% and when focusing just on the change for $n = m + 1$ the increase is less than 30%. The impact is more moderate for higher parameter sets due to other dominating computations that are the same with and without the optimization. For the Level I parameter set, our optimization reduces the signature size to under one quarter. Since the size of w non-seeded responses drops from quadratic to linear in the security parameters ($m^2 + n^2$ to $2k$), we obtain a particularly strong up to over $20\times$ improvement in signature size for the larger parameter sets.

However, for Level I and III, the attack cost actually is reduced when incrementing n by one, based on the attacks described in the MEDS specification [10]. Hence, to maintain the security level, we need to increase m, n, and k by one, which increases all public key size, signature size, and signing time, as shown in the fourth row of the Level I and III parameter sets. Nevertheless, due to the significant reduction of the signature size compared to the original "spec [10]" case, we can compensate for these performance penalties by modifying the performance parameters, i.e., by reducing s (to reduce the public key size), by reducing t (to reduce signing time), and by increasing w correspondingly (which increases the signature size). The result of this is shown in the fifth row for the Level I and III parameter sets and in the fourth row for the Level V parameters.

Overall, our signature optimization allows us to reduce the signature size significantly to about 30% at Level I, to about 12% at Level III, and to about 6% on Level V at little to no negative impact on public key size and signing time (which for smaller parameter sets even can be improved as well depending on what trade-offs are chosen for the performance parameters s, t, and w).

5 Extending Our Technique to ALTEQ

Our technique is designed to apply to group-action Fiat-Shamir signatures using matrix codes. It is however not immediately clear whether it is still a valid approach under specific constraints. In this section we answer positively this question for the case of matrix codes arising from alternating trilinear forms used by ALTEQ [22,6]. In this case the isometries can be represented by a single matrix, and the matrix codes show a specific symmetry, coming from the alternating-ness, not present in random matrix codes.

		q	n	m	k	bit sec.	s	t	w	pk (byte)	sig (byte)	sign (mcyc)
						NIST Security Level I						
Ia 1	spec [10]	4093	14	14	14	146.52	4	1152	14	9 923	9 896	486.89
Ia 2	spec n++		15			130.05†				10 685	10 512	551.40
Ia 3	sig opt		15			130.05†				10 616	2 252	708.81
Ia 4	sig opt		16	15	15	148.61				13 196	2 294	896.85
Ia 5	sig opt		16			148.61	2	256	30	4 420	2 886	199.24
Ib 1	spec [10]	4093	14	14	14	146.52	5	192	20	13 220	12 976	82.34
Ib 2	spec n++		15			130.05†				14 236	13 856	88.38
Ib 3	sig opt		15			130.05†				14 144	2 056	120.43
Ib 4	sig opt		16	15	15	148.61				17 584	2 116	147.61
Ib 5	sig opt		16			148.61	2	144	48	4 420	4 016	112.22
						NIST Security Level III						
IIIa 1	spec [10]	4093	22	22	22	216.83	4	608	26	41 711	41 080	1 244.56
IIIa 2	spec n++		23			195.38†				43 697	42 848	1 325.20
IIIa 3	sig opt		23			195.38†				43 592	5 044	1 730.32
IIIa 4	sig opt		24	23	23	213.64				50 024	5 122	2 027.83
IIIa 5	sig opt		24			213.64	3	480	31	33 360	5 203	1 572.73
IIIb 1	spec [10]	4093	22	22	22	216.83	5	160	36	55 604	54 736	332.91
IIIb 2	spec n++		23			195.38†				58 252	57 184	361.72
IIIb 3	sig opt		23			195.38†				58 112	4 840	450.68
IIIb 4	sig opt		24	23	23	213.64				66 688	4 948	524.62
IIIb 5	sig opt		24			213.64	3	128	69	33 360	6 241	425.93
						NIST Security Level V						
Va 1	spec [10]	2039	30	30	30	291.31	5	192	52	134 180	132 528	1 229.89
Va 2	spec n++		31			298.24				138 804	136 896	1 285.49
Va 3	sig opt		31			298.24				138 632	8 092	1 623.09
Va 4	sig opt		31			298.24	4	144	74	103 982	10 302	1 257.84
Vb 1	spec [10]	2039	30	30	30	291.31	6	112	66	167 717	165 464	726.48
Vb 2	spec n++		31			298.24				173 497	171 008	750.95
Vb 3	sig opt		31			298.24				173 282	7 526	955.19
Vb 4	sig opt		31			298.24	5	112	86	138 632	8 546	955.76

†For these cases, inceasing n to $m+1$ reduces the security under the desired level. To compensate for this, the security paramters need to be increased.

Table 1. Comparison of the $|m-n| \leq 1$ signature size optimization from Section 4.2 ("sig opt") in several variants with the original parameter sets ("spec [10]") and the original parameter sets using $n = m+1$ ("spec n++").

Alternating Trilinear Forms. An alternating trilinear form (ATF) on $\mathbb{F}_q^n$ is a function

$$\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$$

such that it is:

- *Trilinear:* It is linear in each of its three arguments.
- *Alternating:* Whenever $\mathbf{x}_i = \mathbf{x}_j$, for some $i \neq j$, then $\phi(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = 0$.

The vector space of alternating trilinear forms on $\mathbb{F}_q^n$ is denoted by $\bigwedge^3 \mathbb{F}_q^n$. When q is odd, every alternating form is also skew-symmetric, i.e. $\phi(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = -\phi(\mathbf{x}_{\tau(1)}, \mathbf{x}_{\tau(2)}, \mathbf{x}_{\tau(3)})$ for any transposition $\tau \in \mathcal{S}_3$.

A trilinear form is, by trilinearity, completely determined by its value on all triples of basis vectors $c_{ijk} = \phi(\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k)$ for $1 \leq i, j, k \leq n$. If in addition the form is alternating, there is redundancy in these values, since $c_{ijk} = -c_{ikj} = c_{kij} = -c_{jik} = c_{jki} = -c_{kji}$ Therefore, an ATF is completely determined by $c_{ijk} = \phi(\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k)$ for $1 \leq i < j < k \leq n$ and can be represented by $\binom{n}{3}$ field elements.

For a given $\mathbf{v} \in \mathbb{F}_q^n$, we will use the notation $\phi_\mathbf{v} = \phi(-, -, \mathbf{v})$ to denote a partially fixed trilinear form. Similarly, we will write $\phi_i = \phi_{\mathbf{e}_i} = \phi(-, -, \mathbf{e}_i)$ in the case of basis vectors. By abuse of notation we will sometimes write $\phi_\mathbf{v}$ for the skew-symmetric matrix $(\phi_\mathbf{v})_{jk} = \phi_\mathbf{v}(\mathbf{e}_j, \mathbf{e}_k)$. In our optimization we will make use of the following observation:

$$\phi(\mathbf{u}, \mathbf{v}, \mathbf{w}) = [\phi_1(\mathbf{u}, \mathbf{v}), \ldots, \phi_n(\mathbf{u}, \mathbf{v})] \cdot \mathbf{w}, \quad \forall\ \mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{F}_q^n$$

Here $\cdot$ represents the standard inner product. For simplicity we will denote

$$\phi_{\mathbf{u},\mathbf{v}} = [\phi_1(\mathbf{u}, \mathbf{v}), \ldots, \phi_n(\mathbf{u}, \mathbf{v})] \in \mathbb{F}_q^n.$$

Note that alternating-ness implies that $\phi_{\mathbf{u},\mathbf{v}} = \phi_{\mathbf{v},\mathbf{u}}$.

A Cryptographic Group Action. The Alternating Trilinear Form Equivalence problem can be seen as the vectorization problem of a group action of $\mathrm{GL}_n(q)$ on $\bigwedge^3 \mathbb{F}_q^n$. This group action can be constructed as follows, given a linear transformation $\mathbf{A} \in \mathrm{GL}_n(q)$ we can naturally lift this to $\bigwedge^3 \mathbb{F}_q^n$ via

$$\phi \cdot \mathbf{A} = \phi \circ \mathbf{A} = \phi(\mathbf{A}(-), \mathbf{A}(-), \mathbf{A}(-)).$$

In this way we get a right group action of $\mathrm{GL}_n(q)$ on $\bigwedge^3 \mathbb{F}_q^n$. Now, given the group action, we define the Alternating Trilinear Form Equivalence (ATFE) problem:

Problem 2 (Alternating Trilinear Form Equivalence). $\mathsf{ATFE}(n, \phi, \psi)$:
Input: Two alternating trilinear forms ϕ, ψ.
Question: Find – if any – $\mathbf{A} \in \mathrm{GL}_n(q)$ such that $\phi \cdot \mathbf{A} = \psi$ i.e.:

$$\phi(\mathbf{Ax}, \mathbf{Ay}, \mathbf{Az}) = \psi(\mathbf{x}, \mathbf{y}, \mathbf{z}) \quad \forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{F}_q^n.$$

Since ATFE is a hard problem, we obtain a cryptographic group action.

The ATFE problem can be restated in terms of matrix codes. Given an alternating trilinear form ϕ, we denote by $\langle \phi_1, \ldots, \phi_n \rangle$ the code generated by the matrices $\phi_1, \ldots, \phi_n$. We will say that this code is the matrix code representation of ϕ. We now have the following formulation:

Problem 3 (Alternating Trilinear Form Equivalence (Matrix Code formulation)). $\mathsf{ATFE}(n, \phi, \psi)$:
Input: Two alternating trilinear forms in matrix code representation $\langle \phi_1, \ldots, \phi_n \rangle$, $\langle \psi_1, \ldots, \psi_n \rangle$.
Question: Find – if any – $\mathbf{A} \in \mathrm{GL}_n(q)$ such that $\mathbf{A}^\top \phi_i \mathbf{A} \in \langle \psi_1, \ldots, \psi_n \rangle \subset \mathbb{F}_q^{n \times n}$ for all $1 \leq i \leq n$.

This restatements shows that ATFE can be considered as a matrix code equivalence problem with restriction on the type of isometry and the structure of the codes. Note that both MCE and ATFE are TI-complete as shown in [21] and [18].

5.1 The Basic ALTEQ Protocol

The basic ALTEQ protocol is a straightforward application of Fiat-Shamir to an equivalence problem. It is similar to the basic MEDS protocol except that the ATF is represented using its compact representation that requires only $\binom{n}{3}$ entries (as opposed to a matrix code representation that requires n^3 entries). It is given in Figure 6. Note that the basic ALTEQ protocol can be optimized similarly to other group-action Fiat-Shamir signatures using standard techniques such as multiple public keys, fixed-weight challenges and seed-trees (see Section 3). The authors in [6] consider as part of their submission only the first two. They provide an implementation of seed-trees as well, but leave as open their usage for optimization of the signature size.

Remark 1. The ALTEQ specifications [6] considers a left group action instead of a right group action. These two can be easily translated to each other since acting on the right with $\mathbf{A}$ is equivalent to acting on the left with $\mathbf{A}^\top$ and vice-versa. It is clear that choosing either has no impact on the validity or security of the protocol. For the sake of this exposition and for reducing the amount of transpose symbols, we will use the right group action throughout. Therefore we also state the basic ALTEQ protocol with a right group action in Figure 6.

5.2 The Solve Procedure for ALTEQ

The main idea of our technique introduced in the previous sections is to recover the full isometry from a small given part of the isometry itself and matching pairs of codewords from the two isometric codes that the isometry maps one to another. In MEDS, the isometry is the most general one, so it is rather easy to find $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$ from part of $\tilde{\mathbf{T}}$ and two pairs of matching codewords.

In ALTEQ, if we view it in matrix code representation, we have $\tilde{\mathbf{A}} = \tilde{\mathbf{B}} = \tilde{\mathbf{T}}$. Trying to find algebraically the coefficients of $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{A}}^{-1}$ as in MEDS, might

Public Data
$q, n, \lambda \in \mathbb{N}$.
$\mathbf{A}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$.

I. Keygen()
1. $\phi_0 \xleftarrow{\$} \bigwedge^3 \mathbb{F}_q^n$.
2. $\mathbf{A}_1 \xleftarrow{\$} \mathrm{GL}_n(q)$.
3. $\phi_1 \leftarrow \phi_0 \cdot \mathbf{A}_1$.
4. $\mathsf{sk} \leftarrow \mathbf{A}_1$ and $\mathsf{pk} \leftarrow (\phi_0, \phi_1)$.
5. Return $(\mathsf{sk}, \mathsf{pk})$.

II. Commit(pk)
1. $\tilde{\mathbf{A}} \xleftarrow{\$} \mathrm{GL}_n(q)$.
2. $\psi \leftarrow \phi_0 \cdot \tilde{\mathbf{A}}$.
3. $\mathsf{cmt} \leftarrow \psi$.
4. Return cmt.

III. Challenge()
1. $\mathsf{ch} \xleftarrow{\$} \{0, 1\}$. Return ch.

IV. Response(sk, pk, cmt, ch)
1. $\mathsf{rsp} \leftarrow \mathbf{A}_{\mathsf{ch}}^{-1} \cdot \tilde{\mathbf{A}}$.
2. Return rsp.

V. Verify(pk, cmt, ch, rsp)
1. Parse rsp into $\mathbf{A}'$.
2. If $\mathbf{A}' \notin \mathrm{GL}_n(q)$, return "reject".
3. $\mathsf{cmt}' \leftarrow \phi_{\mathsf{ch}} \cdot \mathbf{A}'$.
4. If $\mathsf{cmt}' = \mathsf{cmt}$, return "accept". Otherwise, return "reject".

Fig. 6. ALTEQ Sigma Protocol.

not be the best strategy. The matrices being the same seems to cause a problem and next, we see why. Nevertheless, we show how we can actually make use of it to achieve a similar effect as in MEDS.

Let us fix the first α columns of $\tilde{\mathbf{A}}$. Denote by A_i the ith column of $\tilde{\mathbf{A}}$. Now, since for $i \leq \alpha$, we know that $\tilde{\mathbf{A}}\mathbf{e}_i = A_i$, we can compute the matrices:

$$\phi(\tilde{\mathbf{A}}\mathbf{e}_i, \tilde{\mathbf{A}}(-), \tilde{\mathbf{A}}(-)) = \phi_{A_i}(\tilde{\mathbf{A}}(-), \tilde{\mathbf{A}}(-)), \quad \forall i \leq \alpha.$$

Now, just as in MEDS we can consider these to be codewords, fix matching target codewords in the ephemeral ATF , and build a system of equations to find the rest of the isometry. For example, we can fix a skew-symmetric matrix ψ_i and create a system using the following equality:

$$\phi_{A_i}(\tilde{\mathbf{A}}(-), \tilde{\mathbf{A}}(-)) = \psi_i(-, -).$$

However, as we can see, this expression is quadratic in the $\tilde{\mathbf{A}}$ variables, which would be too inefficient to solve during signing. We could try to take one $\tilde{\mathbf{A}}$ to the other side:

$$\phi_{A_i}(\tilde{\mathbf{A}}(-), -) = \psi_i(-, \tilde{\mathbf{A}}^{-1}(-)).$$

However, to make sure that $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{A}}^{-1}$ are in fact each others inverses, we again need quadratic equations, which is a problem since the final system will likely be inefficient to solve.

What we can do to avoid quadratic equations, is fix one more argument in ϕ. Now we calculate, for $i \neq j$ and $i \leq \alpha$

$$\phi(\tilde{\mathbf{A}}\mathbf{e}_i, \tilde{\mathbf{A}}\mathbf{e}_j, \tilde{\mathbf{A}}(-)) = \phi(A_i, A_j, \tilde{\mathbf{A}}(-)) = \phi_{A_i, A_j} \cdot \tilde{\mathbf{A}}.$$

Now, this is promising, as $\phi_{A_i,A_j} \cdot \tilde{\mathbf{A}}$ is just a linear expression. If we fix a target vector $\bar{\Psi}_{i,j} \in \mathbb{F}_q^n$ to match it with, we should get n equations in the $\tilde{\mathbf{A}}$ variables. Or at least, so it seems: We do have to be careful here, because for $k \leq \alpha$,

$$(\bar{\Psi}_{i,j})_k = (\phi_{A_i,A_j} \cdot \tilde{\mathbf{A}})_k = \phi_{A_i,A_j} \cdot \tilde{\mathbf{A}} \cdot \mathbf{e}_k = \phi_{A_i,A_j} \cdot A_k.$$

But this equation does not contain any $\tilde{\mathbf{A}}$ variables! So it will almost surely be an inconsistent equation. To avoid this from happening, we fix the last $n - \alpha$ positions of the vector $\bar{\Psi}_{i,j}$ as $\Psi_{i,j} \in \mathbb{F}_q^{n-\alpha}$ and consider only the last $n - \alpha$ columns of $\tilde{\mathbf{A}}$, which we denote by $\tilde{\mathbf{A}}_x$, resulting in:

$$\Psi_{i,j} = \phi_{A_i,A_j} \cdot \tilde{\mathbf{A}}_x$$

Now since we fixed αn entries of our isometry, and for every pair $i, j \leq \alpha$ we get $n - \alpha$ equations, we need to choose target vectors for n pairs $i, j \leq \alpha$. Furthermore, by symmetry, $\phi_{A_i,A_j} = \phi_{A_j,A_i}$, so $\Psi_{i,j} = \Psi_{j,i}$. In other words, we need n pairs with $i \leq j \leq \alpha$. This gives the condition $\binom{\alpha}{2} \geq n$ which determines how big must α be.

Let $(i_1, j_1), \ldots, (i_n, j_n)$ be such a list of n distinct pairs with $i_k \leq j_k \leq \alpha$. Then let us construct the following matrices:

$$\Phi_{\tilde{\mathbf{A}}} = \begin{bmatrix} \phi_{A_{i_1},A_{j_1}} \\ \vdots \\ \phi_{A_{i_n},A_{j_n}} \end{bmatrix} \in \mathbb{F}_q^{n \times n} \quad \text{and} \quad \Psi = \begin{bmatrix} \Psi_{i_1,j_1} \\ \vdots \\ \Psi_{i_n,j_n} \end{bmatrix} \in \mathbb{F}_q^{n \times (n-\alpha)}.$$

We can now write our equations as:

$$\Phi_{\tilde{\mathbf{A}}} \cdot \tilde{\mathbf{A}}_x = \Psi.$$

When $\Phi_{\tilde{\mathbf{A}}}$ is invertible, which happens with probability $1/q$, we can compute the remaining values of $\tilde{\mathbf{A}}$ as $\tilde{\mathbf{A}}_x = \Phi_{\tilde{\mathbf{A}}}^{-1} \cdot \Psi$. Since we want an isometry, we only have to check that the entire $\tilde{\mathbf{A}}$ is full rank, which happens with chance $1/q$.

Thus to conclude, if we fix α columns of our isometry and a random matrix $\Psi \in \mathbb{F}_q^{n \times (n-\alpha)}$, then we can create a system in the remaining entries of $\tilde{\mathbf{A}}$ that has at most one solution with chance roughly $2/q$. We summarize the above discussion in the procedure $\mathbf{Solve}_\Psi$ given in Figure 7. In the description, we use an injective function $\mathfrak{I} = (\mathfrak{I}_1, \mathfrak{I}_2) : \{1, \ldots, n\} \to \{(i, j) \mid 1 \leq i < j \leq \alpha\}$ whose purpose is to ease the notation by providing an ordering on the two-dimensional indices relevant in the construction.

5.3 The Optimized Protocol

The above method, following the same reasoning as for MEDS in the previous sections, suggests to execute the protocol using a specific, fixed choice of Ψ for all alternating forms we use. Denote the set of all alternating forms $\phi \in \bigwedge^3 \mathbb{F}_q^n$ with this fixed Ψ by $\mathcal{A}_\Psi$.

Constants: $\Psi \in \mathbb{F}_q^{n\times(n-\alpha)}$, and an injective function $\mathfrak{I} : \{1,\ldots,n\} \to \{(i,j) \mid 1 \le i < j \le \alpha\}$.
Input: $\phi_0 \in \bigwedge^3 \mathbb{F}_q^n$, $A_i \in \mathbb{F}_q^n$ for $i \le \alpha$.
Output: $\tilde{\mathbf{A}} \in \mathrm{GL}_n(q)$ such that
$\tilde{\mathbf{A}}_i = A_i$ for $i \le \alpha$ and $(\phi_0 \cdot \tilde{\mathbf{A}})(\mathbf{e}_{\mathfrak{I}_1(i)}, \mathbf{e}_{\mathfrak{I}_2(i)}, \mathbf{e}_{j+\alpha}) = \Psi_{i,j}$, or $\perp$.

1. Construct $\Phi_{\tilde{\mathbf{A}}} = \{\phi_0\left(A_{\mathfrak{I}_1(i)}, A_{\mathfrak{I}_2(i)}, e_j\right)\}_{i,j}$
2. If $\Phi_{\tilde{\mathbf{A}}} \notin \mathrm{GL}_n(q)$, return $\perp$.
3. $\tilde{\mathbf{A}} = (A_1 \ \cdots \ A_\alpha \mid \Phi_{\tilde{\mathbf{A}}}^{-1}\Psi)$.
4. If $\tilde{\mathbf{A}} \notin \mathrm{GL}_n(q)$, return $\perp$. Otherwise return $\tilde{\mathbf{A}}$.

Fig. 7. The function $\mathbf{Solve}_\Psi$.

To see that restricting the protocol to $\mathcal{A}_\Psi$ does not impact the security we make the following argument.

Let us assume that ATFE restricted to $\mathcal{A}_\Psi$ is an easy problem, i.e. given $\tilde{\phi}, \tilde{\psi} \in \mathcal{A}_\Psi$, we can find an isometry $\tilde{\phi} \xrightarrow{\mathbf{A}} \tilde{\psi}$ efficiently. Now we want to show that with assumption, ATFE is easy as well to obtain a contradiction.

Let $\phi, \psi \in \bigwedge^3 \mathbb{F}_q^n$, then we can, using **Solve**, efficiently find $\tilde{\phi}, \tilde{\psi} \in \mathcal{A}_\Psi$ and $\tilde{\mathbf{A}}, \tilde{\mathbf{B}} \in \mathrm{GL}_n(q)$ with $\phi \xrightarrow{\tilde{\mathbf{A}}} \tilde{\phi}$ and $\psi \xrightarrow{\tilde{\mathbf{B}}} \tilde{\psi}$. But now, using our assumption, we can efficiently find an isometry $\tilde{\phi} \xrightarrow{\mathbf{A}} \tilde{\psi}$. Using this we can construct the isometry $\tilde{\mathbf{B}}^{-1} \circ \mathbf{A} \circ \tilde{\mathbf{A}} : \phi \to \psi$. Hence we could solve ATFE efficiently, which is a contradiction.

To summarize the discussion, we present our optimized protocol in Figure 8. We formally show its security with the following theorem.

Theorem 3. *The protocol in Figure 8 is a Σ-protocol for the* ATFE *relation.*

Proof. We prove as usual the properties of completeness, special soundness and honest-verifier zero-knowledgeness.

Completeness: For $(\mathsf{cmt}_0, \mathsf{rsp}_0)$ being an honestly generated commitment-response pair for $\mathsf{ch} = 0$, rsp_0 consists of the first α columns, $A_1, \ldots, A_\alpha$, of $\tilde{\mathbf{A}}$. This means that $\mathbf{A}'$ is parsed into exactly these columns which results in $\tilde{\mathbf{A}}' = \tilde{\mathbf{A}}$ and $\mathbf{Verify}(\mathsf{pk}, \mathsf{cmt}_0, 0, \mathsf{rsp}_0) =$ "accept".

For ch_1, the response rsp_1 consists of the columns $A'_i = \mathbf{A}_1^{-1} A_i$ for $1 \le i \le \alpha$. Now for $i \le j \le \alpha$ we have the following equality:

$$\begin{aligned}\psi_{A'_i, A'_j} &= \psi(\mathbf{A}_1^{-1}A_i, \mathbf{A}_1^{-1}A_j, \mathbf{A}_1^{-1}\mathbf{A}_1 -) \\ &= \phi(A_i, A_j, \mathbf{A}_1 -) \\ &= \phi_{A_i, A_j} \cdot \mathbf{A}_1.\end{aligned}$$

Public Data

$q, n, \lambda \in \mathbb{N}$.
$\mathbf{A}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$.
$\alpha = \min\{m \mid m \in \mathbb{N}, \binom{m}{2} \geq n\}$.
An injective function
$\mathfrak{I} : \{1, \ldots, n\} \to \{(i,j) | 1 \leq i < j \leq \alpha\}$.
$\Psi \in \mathbb{F}_q^{n \times (n-\alpha)}$.

II. Commit(pk)

1. $A_i \xleftarrow{\$} \mathbb{F}_q^n$ for $1 \leq i \leq \alpha$.
2. $\tilde{\mathbf{A}} \leftarrow \mathbf{Solve}_{\Psi}(\phi_0, \{A_i\})$
3. If $\tilde{\mathbf{A}} = \perp$, restart at 1.
4. $\psi \leftarrow \phi_0 \cdot \tilde{\mathbf{A}}$.
5. $\mathsf{cmt} \leftarrow \psi$.
6. Return cmt.

IV. Response(sk, pk, cmt, ch)

1. $\mathsf{rsp} \leftarrow \mathbf{A}_{\mathsf{ch}}^{-1} \cdot (A_1 \ \cdots \ A_\alpha) \in \mathbb{F}_q^{n \times \alpha}$.
2. Return rsp.

I. Keygen()

1. $\phi_0 \xleftarrow{\$} \mathcal{A}_{\Psi}$.
2. $A_i \xleftarrow{\$} \mathbb{F}_q^n$ for $1 \leq i \leq \alpha$.
3. $\mathbf{A}_1 \leftarrow \mathbf{Solve}_{\Psi}(\phi_0, \{A_i\})$
4. $\phi_1 \leftarrow \phi_0 \cdot \mathbf{A}_1$.
5. $\mathsf{sk} \leftarrow \mathbf{A}_1$ and $\mathsf{pk} \leftarrow (\phi_0, \phi_1)$.
6. Return $(\mathsf{sk}, \mathsf{pk})$.

III. Challenge()

1. $\mathsf{ch} \xleftarrow{\$} \{0,1\}$. Return ch.

V. Verify(pk, cmt, ch, rsp)

1. Parse rsp into $\mathbf{A}' \in \mathbb{F}_q^{n \times \alpha}$.
2. $A'_i \leftarrow \mathbf{A}'\mathbf{e}_i$, for $1 \leq i \leq \alpha$.
3. $\tilde{\mathbf{A}}' \leftarrow \mathbf{Solve}_{\Psi}(\phi_{\mathsf{ch}}, \{A'_i\})$
4. If $\tilde{\mathbf{A}}' = \perp$, return "reject".
5. $\psi \leftarrow \phi_{\mathsf{ch}} \cdot \tilde{\mathbf{A}}'$.
6. $\mathsf{cmt}' \leftarrow \psi$.
7. If $\mathsf{cmt}' = \mathsf{cmt}$, return "accept". Otherwise, return "reject".

Fig. 8. Optimized Σ-protocol for ALTEQ.

Thus, in our **Solve** procedure the matrix $\Phi_{\tilde{\mathbf{A}}'}$ that we build from ψ and A'_is is equal to:

$$\Phi_{\tilde{\mathbf{A}}'} = \begin{bmatrix} \psi_{A'_{i_1}, A'_{j_1}} \\ \vdots \\ \psi_{A'_{i_n}, A'_{j_n}} \end{bmatrix} = \begin{bmatrix} \phi_{A_{i_1}, A_{j_1}} \cdot \mathbf{A}_1 \\ \vdots \\ \phi_{A_{i_n}, A_{j_n}} \cdot \mathbf{A}_1 \end{bmatrix} = \Phi_{\tilde{\mathbf{A}}} \cdot \mathbf{A}_1$$

A first conclusion is that invertibility of $\Phi_{\tilde{\mathbf{A}}}$ is equivalent to invertibility of $\Phi_{\tilde{\mathbf{A}}'}$. Then, when we solve $\Phi_{\tilde{\mathbf{A}}'}\mathbf{A}' = \Psi$, we obtain

$$\mathbf{A}' = \Phi_{\tilde{\mathbf{A}}'}^{-1}\Psi = \mathbf{A}_1^{-1}\Phi_{\tilde{\mathbf{A}}}^{-1}\Psi = \mathbf{A}_1^{-1}\tilde{\mathbf{A}}.$$

In other words $\mathbf{A}' = \mathbf{A}_1^{-1}\tilde{\mathbf{A}}$ proving completeness.

Zero-knowledge: The proof for Zero-knowledge again follows a similar structure as the ones for the optimized MEDS protocols. The case of ch_0 is straightforward to simulate since the honest procedure for obtaining the pair $(\mathsf{cmt}_0, \mathsf{rsp}_0)$ does not involve the secret key. Because of completeness, we can immediately conclude that running the honest commit procedure without the secret key but instead of ϕ, using ψ produces a valid transcript $(\mathsf{cmt}_1, \mathsf{ch}_1, \mathsf{rsp}_1)$. Now since we can sample from $\{\tilde{\mathbf{A}} \in \mathrm{GL}_n(q) \mid \psi \cdot \tilde{\mathbf{A}} \in \mathcal{A}_{\Psi}\}$ uniformly random and the ephemeral isometry is sampled uniformly random from $\{\tilde{\mathbf{A}} \in \mathrm{GL}_n(q) \mid \phi \cdot \tilde{\mathbf{A}} \in \mathcal{A}_{\Psi}\}$, it follows that distribution of the transcripts of the honest and simulated protocol are the same, so we have zero-knowledge.

Special-soundness: The premise of the optimization of the protocol is to send isometries with less information. Since one can construct the respective isometry from each response, special soundness is evident.

5.4 New Sizes

For the comparison of the new sizes to the original protocol we take the same conventions. We denote by C the amount of public keys (excluding ϕ_0), by r the number of rounds in the Fiat-Shamir construction, and by K the amount of rounds with non-zero challenge. Furthermore we denote by λ the security parameter and the size of our seeds. Now the security of the Fiat-Shamir construction can still be calculated as:

$$\binom{r}{K} \cdot C^K$$

The new public key-size and signature size can be computed as:

$$\text{PubKeySize} = C\left(\binom{n}{3} - n(n-\alpha)\right) \cdot \lceil \log_2 q \rceil + \lambda$$
$$\text{SigSize} = (r - K + 2) \cdot \lambda + K \cdot \alpha \cdot n \cdot \lceil \log_2 q \rceil$$

For NIST Security Level I, III, and V we take λ to be 128, 192 and 256 respectively, in line with the original ALTEQ specification.

Since the sizes of the public keys and isometries have changed, it can be worthwhile to re-optimize the Fiat-Shamir parameters. For the balanced parameters we optimized on $\mathsf{pk} + \mathsf{sig}$. For the ShortSig parameter sets we limited the public key to be of size 512 KB, 1 MB, and 2 MB respectively, in line with the original specs. These re-optimizations of the Fiat-Shamir parameters are indicated by a "+"-sign. In the re-optimizations we did not take computational speed into account. One could always limit the number of rounds and re-optimize.

These size reductions come at a small computational cost. We call C_{iso} the cost of a group action computation, C_{multi} the cost of a matrix multiplication and C_{inverse} the cost of matrix inversion then we can state the costs of the original protocol as follows:

$$C_{\textbf{Keygen}} = C \cdot C_{\text{iso}}$$
$$C_{\textbf{Commit}} = r \cdot C_{\text{iso}} + K \cdot C_{\text{multi}}$$
$$C_{\textbf{Verify}} = r \cdot C_{\text{iso}}$$

In the optimized protocol we have the added computational cost of **Solve**. If we assume that computing $\Phi_{\tilde{\mathbf{A}}}$ in **Solve** is as costly as C_{iso} then we can state its computational costs as follows:

$$C'_{\textbf{Keygen}} = C \cdot (2 \cdot C_{\text{iso}} + C_{\text{multi}} + C_{\text{inverse}})$$
$$C'_{\textbf{Commit}} = r \cdot (2 \cdot C_{\text{iso}} + C_{\text{multi}} + C_{\text{inverse}}) + K \cdot C_{\text{multi}}$$
$$C'_{\textbf{Verify}} = r \cdot (2 \cdot C_{\text{iso}} + C_{\text{multi}} + C_{\text{inverse}})$$

This is a small price for the reduction in signature size.

		n	α	C	r	K	pk (byte)	sig (byte)	pk + sig (byte)
				NIST Security Level I					
Balanced	spec [6]	13	–	7	84	22	8 024	15 896	23 920
Balanced	optimized	13	6	7	84	22	5 476	7 888	13 364
Balanced+	optimized	13	6	3	160	23	2 356	9 400	11 756
				NIST Security Level III					
Balanced	spec [6]	20	–	7	201	28	31 944	49 000	80 944
Balanced	optimized	20	7	7	201	28	24 664	19 880	44 544
Balanced+	optimized	20	7	2	306	36	7 064	26 688	33 752
				NIST Security Level V					
Balanced	spec [6]	25	–	8	119	48	73 632	122 336	195 968
Balanced	optimized	25	8	8	119	48	60 032	40 736	100 768
Balanced+	optimized	25	8	2	424	47	15 032	49 728	64 760

Table 2. Comparison of the public key and signature sizes after optimization using the "Balanced" parameter sets from ("spec [6]").

		n	α	C	r	K	pk (byte)	sig (byte)
			NIST Security Level I					
ShortSig	spec [6]	13	–	458	16	14	523 968	9 528
ShortSig	optimized	13	6	458	16	14	357 256	4 432
ShortSig+	optimized	13	6	657	29	11	512 476	3 752
			NIST Security Level III					
ShortSig	spec [6]	20	–	229	39	20	1 044 264	32 504
ShortSig	optimized	20	7	229	39	20	806 104	11 704
ShortSig+	optimized	20	7	297	69	17	1 045 464	10 816
			NIST Security Level V					
ShortSig	spec [6]	25	–	227	67	25	2 088 432	63 908
ShortSig	optimized	25	8	227	67	25	1 702 532	21 408
ShortSig+	optimized	25	8	276	88	23	2 070 032	20 544

Table 3. Comparison of the public key and signature sizes after optimization using the "ShortSig" parameter sets from ("spec [6]").

References

1. NIST additioal signature round announcement. NIST Official Website (2021), https://csrc.nist.gov/projects/pqc-dig-sig
2. 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
3. Barenghi, A., Biasse, J.F., Persichetti, E., Santini, P.: LESS-FM: Fine-tuning Signatures from the Code Equivalence Problem. In: International Conference on Post-Quantum Cryptography. pp. 23–43. Springer (2021)
4. Beullens, W.: Breaking Rainbow Takes a Weekend on a Laptop. In: CRYPTO. Lecture Notes in Computer Science, vol. 13508, pp. 464–479. Springer (2022)
5. Beullens, W., Katsumata, S., Pintore, F.: Calamari and Falafl: logarithmic (linkable) ring signatures from isogenies and lattices. In: ASIACRYPT 2020. pp. 464–492. Springer (2020)
6. Bläser, M., Duong, D.H., Narayanan, A.K., Plantard, T., Qiao, Y., Sipasseuth, A., Tang, G.: The alteq signature scheme: Algorithm specifications and supporting documentation (2023), https://pqcalteq.github.io/ALTEQ_spec_2023.09.18.pdf
7. Bouillaguet, C., Fouque, P., Véber, A.: Graph-theoretic algorithms for the "isomorphism of polynomials" problem. In: Johansson, T., Nguyen, P.Q. (eds.) Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7881, pp. 211–227. Springer (2013)
8. Boyar, J., Erfurth, S., Larsen, K.S., Niederhagen, R.: Quotable signatures for authenticating shared quotes. In: Aly, A., Tibouchi, M. (eds.) Progress in Cryptology – LATINCRYPT 2023. LNCS, vol. 14168, pp. 273–292. Springer (2023)
9. Brent, R.P., Gao, S., Lauder, A.G.: Random krylov spaces over finite fields. SIAM Journal on Discrete Mathematics **16**(2), 276–287 (2003)
10. Chou, T., Niederhagen, R., Persichetti, E., Ran, L., Randrianarisoa, T.H., Reijnders, K., Samardjiska, S., Trimoska, M.: MEDS – Matrix Equivalence Digital Signature (2023), https://meds-pqc.org/spec/MEDS-2023-05-31.pdf, submission to the NIST Digital Signature Scheme standardization process.
11. 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. LNCS, vol. 14064, pp. 28–52. Springer (2023)
12. Chou, T., Persichetti, E., Santini, P.: On linear equivalence, canonical forms, and digital signatures (2023), https://eprint.iacr.org/2023/1533.pdf
13. De Feo, L., Galbraith, S.D.: SeaSign: Compact Isogeny Signatures from Class Group Actions. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. pp. 759–789. Springer International Publishing (2019)
14. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: CRYPTO. pp. 186–194. Springer (1986)
15. Fulman, J.: Random matrix theory over finite fields. Bulletin of the American Mathematical Society **39**(1), 51–85 (2002)
16. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM **33**(4), 792–807 (Aug 1986)

17. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. J. ACM **38**(3), 690–728 (jul 1991). https://doi.org/10.1145/116825.116852, https://doi.org/10.1145/116825.116852
18. Grochow, J.A., Qiao, Y., Tang, G.: Average-case algorithms for testing isomorphism of polynomials, algebras, and multilinear forms. Journal of Groups, Complexity, Cryptology **Volume 14, Issue 1** (Aug 2022). https://doi.org/10.46298/jgcc.2022.14.1.9431, https://gcc.episciences.org/9836, preliminary version appeared in STACS '21, doi:10.4230/LIPIcs.STACS.2021.38. Preprint available at arXiv:2012.01085
19. Kipnis, A., Patarin, J., Goubin, L.: Unbalanced Oil and Vinegar Signature Schemes. In: EUROCRYPT '99. Lecture Notes in Computer Science, vol. 1592, pp. 206–222. Springer (1999)
20. National Institute for Standards and Technology: Post-Quantum Cryptography Standardization (2017), https://csrc.nist.gov/Projects/Post-Quantum-Cryptography
21. Reijnders, K., Samardjiska, S., Trimoska, M.: Hardness estimates of the Code Equivalence Problem in the Rank Metric. Des. Codes Cryptogr. (2024). https://doi.org/https://doi.org/10.1007/s10623-023-01338-x
22. Tang, G., Duong, D.H., Joux, A., Plantard, T., Qiao, Y., Susilo, W.: Practical post-quantum signature schemes from isomorphism problems of trilinear forms. In: EUROCRYPT 2022. Lecture Notes in Computer Science, vol. 13277, pp. 582–612. Springer (2022)
23. Tao, C., Petzoldt, A., Ding, J.: Efficient Key Recovery for All HFE Signature Variants. In: CRYPTO (1). Lecture Notes in Computer Science, vol. 12825, pp. 70–93. Springer (2021)