Zero Knowledge Protocols and Signatures from the Restricted Syndrome Decoding Problem

Marco Baldi[1], Sebastian Bitzer[2], Alessio Pavoni[1], Paolo Santini[1], Antonia Wachter-Zeh[2], and Violetta Weger[2]

[1] Marche Polytechnic University, Italy
Department of Information Engineering
{m.baldi, p.santini} @staff.univpm.it, s1101018@studenti.univpm.it
[2] Technical University of Munich, Germany,
TUM School of Computation, Information and Technology
{sebastian.bitzer, antonia.wachter-zeh, violetta.weger}@tum.de

Abstract. The Restricted Syndrome Decoding Problem (R-SDP) corresponds to the Syndrome Decoding Problem (SDP) with the additional constraint that entries of the solution vector must live in a desired subset of a finite field. In this paper we study how this problem can be applied to the construction of signatures derived from Zero-Knowledge (ZK) proofs. First, we show that R-SDP appears to be well suited for this type of applications: almost all ZK protocols relying on SDP can be modified to use R-SDP, with important reductions in the communication cost. Then, we describe how R-SDP can be further specialized, so that solutions can be represented with a number of bits that is slightly larger than the security parameter (which clearly provides an ultimate lower bound), thus enabling the design of ZK protocols with tighter and rather competitive parameters. Finally, we show that existing ZK protocols can greatly benefit from the use of R-SDP, achieving signature sizes in the order of 7 kB, which are smaller than those of several other schemes obtained from ZK protocols. For instance, this beats all schemes based on the Permuted Kernel Problem (PKP), almost all schemes based on SDP and several schemes based on rank metric problems.

1 Introduction

NIST's announcement for an additional round of the post-quantum cryptography competition, devoted to signatures based on non-structured lattices, has undoubtedly motivated the cryptographic community in studying new and efficient schemes. As a matter of fact, the panorama has greatly changed in the last couple of years. In particular, significant effort has been dedicated to schemes that are obtained by applying the Fiat-Shamir transform to a Zero Knowledge (ZK) interactive protocol; nowadays, they are deemed as one of the most efficient solutions to design a post-quantum signature scheme.

Signatures derived from this approach normally have very compact public keys and enjoy high security guarantees, since the proof of knowledge is constructed around a truly random instance of some hard problem (i.e., no trapdoor is required) and the security of the protocol relies on fundamental cryptographic tools, e.g., commitment schemes. This is a crucial difference with signatures stemming from the hash&sign paradigm, which seems difficult to apply to some post-quantum trapdoors, like code-based ones. The most remarkable example is, arguably, that of the Syndrome Decoding Problem (SDP): several hash&sign schemes have been proposed but they require ad-hoc trapdoors to circumvent the technical difficulties of hashing into a decodable syndrome (which normally happens with negligible probability, unless extreme code parameters are chosen). As a matter of fact, all proposed solutions have either security concerns or unpractical parameters (or both) [21, 30, 32, 31, 19, 3]. The only remarkable exception is WAVE [18], which, however, is not based on the canonical SDP setting (the solution vector is not unique and is required to have large weight) and, as a major drawback, has very large public keys (in the order of 3 MB for 128 bits of security).

Historical examples of ZK protocols based on well known and studied post-quantum problems are [35, 37, 17, 1, 33]. However, all of them suffer from rather large signatures, owing to the need of several parallel executions (repetitions of the underlying protocol) to reduce the overall soundness error. For this reason, all the aforementioned schemes have been deemed as impractical, for several years. However, researchers have begun to study new protocols, employing techniques to reduce both the soundness error and the signature size and, in recent years, significant improvements have been made [12, 15, 26, 23, 24, 2, 16, 22, 14, 13]. All of these modern solutions can be thought of as highly optimized versions of historical protocols, and the improvements are due to more efficient ZK protocols with reduced soundness error, without modifications in the underlying hard problem. Popular techniques of this type are the so-called protocol-with-helper, first formalized in [13], and the Multi Party Computation (MPC) in-the-head (MPCItH) paradigm, first proposed in [27]. To contextualize: while historical schemes have signatures of dozens of kB for 128 bits of security, modern approaches are able to bring the signature size down to $5 \div 17$ kB.

1.1 Our contribution

We continue along this line of research and consider how the Restricted Syndrome Problem (R-SDP), introduced in [5][3], can be used to design efficient ZK protocols and signatures. The problem is the same as in the SDP, with the additional constraint that the entries of the solution vector must live in a subset of the underlying finite field, and is NP-complete for any choice of the subset.

First, we argue that R-SDP seems to be more suited for designing proof-of-knowledge systems, with respect to SDP, since existing protocols can more conveniently use R-SDP, with significant reductions in the communication cost. This is motivated by the fact that in R-SDP the error can have a much larger Hamming weight than in SDP, even maximum (that is, no null entry), while

[3] A similar idea was already mentioned in [36], but it was not used in conjunction with a decoding problem.

still having a unique solution to the problem[4]. This increases the cost of generic decoders, as such algorithms are usually searching for many zero entries and enumerate non-zero entries. As a matter of fact, with R-SDP, we can achieve the same security level using smaller codes, and this impacts positively on the communication cost.

Another important improvement is due to the transformations used in ZK protocols. For the classical SDP, they are given by a monomial transformation (a permutation with scaling factors), as these are the transitive linear isometries on the Hamming sphere. For the new R-SDP, diagonal matrices with restricted entries are enough, since they act as a linear transitive map between the restricted full-weight vectors. This yields another significant reduction in the communication cost and, furthermore, simplifies implementations (as constant time implementation of permutations is non-trivial).

Then, we derive a special version of R-SDP, called R-SDP(G), which enables signatures that are as compact as possible. Namely, for a security of λ bits, R-SDP(G) can use a solution space G of size $2^{(1+\alpha)\lambda}$, where $\alpha \geq 0$ is a small constant (say, $\alpha \leq 1$). From a mathematical point of view, G is a group which acts transitively and freely on itself: this implies that we can sample any restricted object (i.e., secret keys and hiding transformations) from G, and achieve a representation using only $(1+\alpha)\lambda \leq 2\lambda$ bits. The value of α is chosen from a conservative perspective, so that existing attacks cannot be sped-up taking into account the knowledge of G.

Finally, we introduce variants of modern ZK protocols relying on R-SDP, and derive the corresponding signature sizes. In particular, we focus on the GPS scheme for SDP [26] and BG scheme for the Permuted Kernel Problem (PKP) [14]. Notice that GPS is based on the protocol-with-helper paradigm, while BG removes the need for the helper and uses shared permutations to reduce the soundness error (a similar technique has been applied also to the context of SDP in [23]). We convert these protocols to the use of R-SDP and R-SDP(G) and derive the signature sizes of the resulting schemes, which we call R-GPS and R-BG[5]. For R-GPS, we almost halve the signature sizes, even when considering the less powerful R-SDP. Also for R-BG we achieve important savings: using R-SDP(G), we obtain signatures with a size of 7.7 kB (instead of 10.0) for the fast variant, and 7.2 kB (instead of 8.9) for the short variant. These results show that R-SDP can be used to design efficient protocols, regardless of the protocol structure (e.g., whether the helper is used or not). We also foresee that R-SDP can be used in combination with MPCItH techniques, but the modifications are not trivial (as we discuss next), and are left for future works.

1.2 Paper organization

Section 2 settles the notation we use and gives (minimal) preliminaries about linear codes and ZK protocols. In Section 3 we formally introduce R-SDP, show

[4] Notice that this is different to the setting considered in WAVE [18], where there are many large weight solutions.

[5] The initial R stands for the fact that the scheme uses R-SDP.

how it can be solved using Information Set Decoding (ISD) and that R-SDP can be much harder than SDP. In Section 4 we show how R-SDP can be applied to the context of ZK protocols, and argue why its use is expected to lead to very promising schemes. This motivates the analysis in Section 5, in which we formally introduce R-SDP(G) and analyze its security. Finally, in Section 6, we describe and analyze the R-GPS and R-BG signature schemes, while Section 7 concludes the paper.

2 Notation and preliminaries

We use $[a; b]$ to denote the set of all reals $x \in \mathbb{R}$ such that $a \leq x \leq b$. For a finite set A, the expression $a \overset{\$}{\leftarrow} A$ means that a is chosen uniformly at random from A. In addition, we denote by $|A|$ the cardinality of A, by A^C its complement and by $A_0 = A \cup \{0\}$. As usual, for q being a positive integer, we denote by $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ the ring of integers modulo q. For q being a prime power, we denote by $\mathbb{F}_q$ the finite field of order q and by $\mathbb{F}_q^*$ its multiplicative group. For $g \in \mathbb{F}_q^*$, we denote by $\mathrm{ord}(g)$ its multiplicative order.

We use uppercase (resp. lowercase) letters to indicate matrices (resp. vectors). If J is a set, we use $\mathbf{A}_J$ to denote the matrix formed by the columns of $\mathbf{A}$ that are indexed by J; analogous notation will be used for vectors. The identity matrix of size m is denoted as $\mathbf{I}_m$. We use $\mathbf{0}$ to denote both the null matrix or the null vector without specifying dimensions (which will always be clear from the context). Finally, we denote by h_q the q-ary entropy function.

2.1 Cryptographic tools

Throughout the paper, we adopt conventional cryptographic notations, e.g., λ denotes the security parameter. Standard functions are always implicitly defined, e.g., Hash will indicate a secure hash function with digests of size 2λ. We focus on Zero-Knowledge (ZK) protocols, that is, interactive protocols in which a *prover* aims to convince a *verifier* that she knows a secret that verifies some public statement, without revealing said secret. In our context, the secret will correspond to the secret key and the public statement to the public key. A protocol is called an N-pass protocol if the number of messages that are exchanged during an execution is N.

Informally, a protocol achieves the ZK property when the interaction between the two parties does not reveal any useful information about the secret held by the prover. We say that a protocol has *soundness error* ε if a cheating prover, i.e., someone that does not know the secret, can convince the honest verifier with probability ε. When t parallel repetitions of a N-pass protocol with soundness error ε are considered, we obtain a new N-pass protocol with soundness error ε^t. Due to lack of space, we do not recall formal definitions for these (and other) properties of ZK protocols, we refer the interested reader to [23].

ZK protocols can be used to obtain a signature schemes using the Fiat-Shamir transform [25]. To achieve a security of λ bits, it is important that the underlying protocol has a soundness error $\varepsilon < 2^{-\lambda}$. When transforming a 5-pass protocol into a signature scheme, some caution is needed thanks to the attack in [28]. When needed, we will discuss how to choose t so that this attack is mitigated.

2.2 Linear codes

A *linear code* $\mathscr{C}$ over the finite field $\mathbb{F}_q$ with length n and dimension $k \leq n$ is a k-dimensional linear subspace of $\mathbb{F}_q^n$. A compact representation for a code is a *generator matrix*, that is, a full rank $\mathbf{G} \in \mathbb{F}_q^{k\times n}$ such that $\mathscr{C} = \left\{\mathbf{uG} \mid \mathbf{u} \in \mathbb{F}_q^k\right\}$. We say that a code of length n and dimension k has *rate* $R = \frac{k}{n}$ and *redundancy* $r = n - k$. Equivalently, one can represent a code through a full rank $\mathbf{H} \in \mathbb{F}_q^{r\times n}$, called *parity-check matrix*, such that $\mathscr{C} = \left\{\mathbf{c} \in \mathbb{F}_q^n \mid \mathbf{cH}^\top = \mathbf{0}\right\}$. The *syndrome* of some $\mathbf{x} \in \mathbb{F}_q^n$ is the length-r vector $\mathbf{s} = \mathbf{xH}^\top$. A set $J \subseteq \{1, \cdots, n\}$ of size k is called *information set* for $\mathscr{C}$ if $|\mathscr{C}_J| = q^k$, where $\mathscr{C}_J = \{\mathbf{c}_J \mid \mathbf{c} \in \mathscr{C}\}$. It easily follows, that $\mathbf{G}_J$ and $\mathbf{H}_{J^C}$ are invertible matrices. We say that a generator matrix, respectively a parity-check matrix is in systematic form (with respect to the information set J), if $\mathbf{G}_J = \mathbf{I}_k$, respectively $\mathbf{H}_{J^C} = \mathbf{I}_{n-k}$. We endow the vector space $\mathbb{F}_q^n$ with the *Hamming metric*: given $\mathbf{x} \in \mathbb{F}_q^n$, its Hamming weight $\mathrm{wt}(\mathbf{x})$ is the number of non-zero entries. The *minimum distance* of a linear code is given by $d = \min\{\mathrm{wt}(\mathbf{c}) \mid \mathbf{c} \in \mathscr{C}, \mathbf{c} \neq \mathbf{0}\}$. Recall that the Gilbert-Varshamov (GV) bound states that $R \geq 1 - h_q(d/n)$. It is well known, that a random code attains the Gilbert-Varshamov bound, for large enough length n, i.e., for a random code we may assume $d/n = h_q^{-1}(1 - R)$. Code-based cryptography usually relies on the following NP-complete problem [10, 7].

Problem 1 Syndrome Decoding Problem (SDP)
Given $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}, t \in \mathbb{N}, \mathbf{s} \in \mathbb{F}_q^{n-k}$*, decide if there exists* $\mathbf{e} \in \mathbb{F}_q^n$*, such that* $wt(\mathbf{e}) \leq t$ *and* $\mathbf{eH}^\top = \mathbf{s}$?

We usually assume that the instance of the SDP is chosen uniformly at random, thus also that the code with parity-check matrix $\mathbf{H}$ attains the GV bound. If the target weight t is less than the minimum distance δn of the GV bound, we expect to have on average a unique solution (if we have any), since (on average) the number of solutions is given by $q^{n(h_q(\delta)-1+R)} \leq 1$.

3 The Restricted Syndrome Decoding Problem

Let us consider some subset $\mathbb{E}$ of $\mathbb{F}_q^\star$, denote by $\mathbb{E}_0 = \mathbb{E} \cup \{0\}$ and by

$$\mathscr{S}_w^{\mathbb{E}} \coloneqq \{\mathbf{x} \in \mathbb{E}_0^n \mid \mathrm{wt}(\mathbf{x}) = w\}$$

the *Hamming sphere with radius w and restriction* $\mathbb{E}$. Clearly, for $\mathbb{E}$ of size z, we have $\mid \mathscr{S}_w^{\mathbb{E}} \mid= \binom{n}{w} z^w$. The Restricted Syndrome Decoding Problem (R-SDP), firstly introduced in [5], reads as follows.

Problem 2 Restricted Syndrome Decoding Problem (R-SDP)
Given $\mathbf{H} \in \mathbb{F}_q^{r\times n}$*,* $\mathbf{s} \in \mathbb{F}_q^r$ *and* $w \in \mathbb{N}$*, decide if there exists* $\mathbf{e} \in \mathcal{S}_w^{\mathbb{E}}$*, such that* $\mathbf{e}\mathbf{H}^\top = \mathbf{s}$.

It is easy to see that R-SDP is strongly related to other well-known hard problems. For instance, when $\mathbb{E} = \mathbb{F}_q^*$, the R-SDP corresponds to the canonical SDP and if $\mathbb{E} = \{1\}$, the R-SDP is similar to the Subset Sum Problem (SSP) over finite fields. Consequently, it is not surprising that R-SDP is NP-complete for any choice of $\mathbb{E}$. The proof is essentially the same as in [5], where the authors focus on the case $\mathbb{E} = \{\pm x_1, \pm x_2, \cdots, \pm x_a\}$. Another proof can be immediately obtained whenever $1 \in \mathbb{E}$ using the proof in [38].

We always consider that the R-SDP instance is chosen uniformly at random. We expect to have on average (at most) a unique solution, if w is such that

$$\binom{n}{w} z^w q^{k-n} \leq 1.$$

Considering this asymptotically, we have for $W = w/n$ the condition

$$2^{n(h_2(W)+W\log_2(z)-(1-R)\log_2(q))} \leq 1,$$

which translates to $\mu(W,z) := W\log_2(z) + h_2(W) - (1-R)\log_2(q) \leq 0$. Let W^* be the maximum value of W for which a random instance of R-SDP is expected to have a unique solution, that is

$$W^* = \max\left\{W \in [0;1] \mid \mu(W,z) \leq 0\right\}. \quad (1)$$

Comparing this to the GV bound, we can see that with the R-SDP, we are allowed to choose a much larger weight w and still guarantee the uniqueness of the solution. This is a crucial difference with SDP, since a high Hamming weight corresponds to an exponentially large number of solutions [18]. Note that if $\log_2(z) \leq (1-R)\log_2(q)$, we even have uniqueness for full-weight vectors.

3.1 Solving R-SDP

To compare the computational complexity of R-SDP with classical SDP, we provide an adaption of the Stern/Dumer algorithm [34, 20], which works for any choice for $\mathbb{E}$. There can be improvements, which depend specifically on the choice and structure of $\mathbb{E}$, for this we refer to Appendix A.

Although the Stern/Dumer algorithm is well-known, we will provide the details in the following for the sake of completeness. As a first step, we choose a set $J \subset \{1,\dots,n\}$ of size $k+\ell$ which contains an information set and perform a Partial Gaussian Elimination (PGE) on $\mathbf{H}$ in the columns indexed by J, obtaining $\widetilde{\mathbf{H}}$, and perform the same operations on the syndrome. For simplicity, let us assume that J is chosen in the first $k+\ell$ positions, thus

$$\mathbf{e}\widetilde{\mathbf{H}}^\top = (\mathbf{e}_1, \mathbf{e}_2)\begin{pmatrix} \mathbf{H}_1^\top & \mathbf{H}_2^\top \\ \mathbf{I}_{r-\ell} & \mathbf{0} \end{pmatrix} = (\mathbf{s}_1, \mathbf{s}_2),$$

where $\mathbf{e}_1 \in \mathbb{E}_0^{k+\ell}, \mathbf{e}_2 \in \mathbb{E}_0^{r-\ell}, \mathbf{H}_1 \in \mathbb{F}_q^{(r-\ell)\times(k+\ell)}, \mathbf{H}_2 \in \mathbb{F}_q^{\ell\times(k+\ell)}, \mathbf{s}_1 \in \mathbb{F}_q^{r-\ell}$ and $\mathbf{s}_2 \in \mathbb{F}_q^{\ell}$. Thus, we get two syndrome equations, being

$$\mathbf{e}_1\mathbf{H}_1^\top + \mathbf{e}_2 = \mathbf{s}_1, \quad \mathbf{e}_1\mathbf{H}_2^\top = \mathbf{s}_2.$$

We solve these equations by requiring that $\mathbf{e}_1$ has weight v; after we find all such $\mathbf{e}_1$, it is enough to check that $\mathbf{s}_1 - \mathbf{e}_1\mathbf{H}_1$ has the remaining weight $w - v$. To solve the smaller instance given by $\mathbf{H}_2, \mathbf{s}_2$ and v, we use a collision search.

To improve readability, we drop any rounding operations and implicitly assume that all parameters can be chosen as integers. For this, we write $\mathbf{e}_1 = (\mathbf{x}_1, \mathbf{x}_2)$ with $\mathbf{x}_i$ of length $(k+\ell)/2$ and weight $v/2$. Thus, we also split $\mathbf{H}_2 = (\mathbf{A}_1, \mathbf{A}_2)$ and construct the two lists

$$\begin{aligned}\mathscr{L}_1 &:= \{(\mathbf{x}_1, \mathbf{x}_1\mathbf{A}_1^\top) \mid \mathbf{x}_1 \in \mathbb{E}_0^{(k+\ell)/2}, \mathrm{wt}(\mathbf{x}_1) = v/2\},\\ \mathscr{L}_2 &:= \{(\mathbf{x}_2, \mathbf{s}_2 - \mathbf{x}_2\mathbf{A}_2^\top) \mid \mathbf{x}_2 \in \mathbb{E}_0^{k+\ell}, \mathrm{wt}(\mathbf{x}_2) = v/2\}.\end{aligned}$$

We then check for collisions, that is, all pairs $(\mathbf{x}_1, \mathbf{a}) \in \mathscr{L}_1$, $(\mathbf{x}_2, \mathbf{a}) \in \mathscr{L}_2$. The two lists are of size $\binom{(k+\ell)/2}{v/2} z^{v/2}$ and the collision search costs approximately $\binom{k+\ell}{v} z^v q^{-\ell}$. The cost of this restricted Stern/Dumer algorithm is then given by

$$\binom{n}{w}\binom{k+\ell}{v}^{-1}\binom{r-\ell}{w-v}^{-1} \cdot \left(\binom{(k+\ell)/2}{v/2} z^{v/2} + \binom{k+\ell}{v} z^v q^{-\ell}\right).$$

Remark 1. Let us denote $L = \ell/n$. In the case of $w = n$, the optimized cost of Stern's algorithm is in $O\big(2^{F(R,q,z)n}\big)$, where

$$F(R,q,z) = \min_{0 \le L \le 1-R} \left\{ \left(\tfrac{R+L}{2}\right)\log_2(z),\ (R+L)\log_2(z) - L\log_2(q) \right\}.$$

In Figure 1 we give the cost of Stern's algorithm for random R-SDP instances, where we choose $W = W^*$, i.e., the maximal weight that guarantees uniqueness. Note that the cost at the point $z = q-1$ corresponds to the cost of Stern on a random SDP instance and thus, we can see that R-SDP with $z < q-1$ has a much larger cost than the SDP with the same parameters q, n, R.

In Appendix A, we solve the R-SDP for the later proposed particular choices of $\mathbb{E}$, by adapting the BJMM algorithm [9] and applying the technique for subset sum solvers of [8]. We observe that the BJMM algorithm can achieve relevant improvements over Stern in the low-weight regime or when the the error set $\mathbb{E}$ possesses a sufficient amount of of additive structure.

The security of R-SDP highly depends on the exact shape of $\mathbb{E}$. There are, indeed, several choices which lead to a somewhat easier problem. For instance, one can choose an extension field $\mathbb{F}_{p^m}$, for some prime p and integer m and $\mathbb{E} \subset \mathbb{F}_{p^m}^\star$. For several choices of $\mathbb{E}$ one can consider the given instance over a subfield and solve an easier problem. For example, if $\mathbb{E} = \mathbb{F}_p^\star$. To avoid this possibility, we directly restrict our considerations to prime fields.

As another suboptimal choice, one can choose rather large values for q and $\mathbb{E} = \{0, 1\}$. Thus, solvers for subset sum problems may be used [8], where one

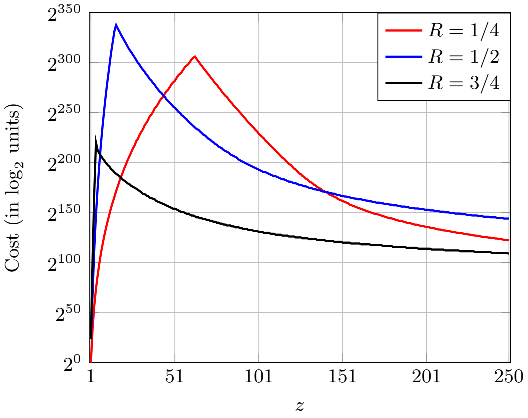

Fig. 1: Cost of Stern's algorithm for random R-SDP instances with $q = 251$, $n = 256$ and several code rate values.

adds some elements to the search space. To circumvent possible speedups from such techniques, we restrict ourselves to error sets $\mathbb{E}$ of relatively large size. For more details on safe choices of $\mathbb{E}$ and attacks using [8], we refer to Appendix A.

4 Building ZK-ID schemes from the R-SDP: a preliminary analysis

In this section we describe how ZK-ID protocols schemes can be obtained from R-SDP. For the majority of this section, we will not consider the MPCItH setting; the discussion on this type of ZK protocols is postponed to the end of the section.

4.1 Zero knowledge masking of restricted vectors

In the schemes we consider in this paper, to achieve zero knowledge it is fundamental that one identifies a set X, from which the secret key is selected, and a group $\mathscr{T}$ that acts transitively on X, with the property that

$$\forall x \in X,\ \sigma(x) \text{ is uniformly distributed over } X \text{ when } \sigma \xleftarrow{\$} \mathscr{T}. \tag{2}$$

As a consequence, when x is the secret key, revealing $y = \sigma(x)$ without revealing σ leaks no information about x.

For schemes based on the SDP, (2) is satisfied choosing X as the Hamming sphere with some radius w and $\mathscr{T}$ as the set of linear isometries, i.e., the set of monomial transformations. A monomial transformation can be described as $(\pi, \mathbf{v})$ with $\pi \in S_n$ a permutation and $\mathbf{v} \in (\mathbb{F}_q^*)^n$. The action of $\sigma = (\pi, \mathbf{v})$ on a vector $\mathbf{a} = (a_1, \cdots, a_n) \in \mathbb{F}_q^n$ corresponds to

$$\sigma(\mathbf{a}) = \big(v_1 a_{\pi(1)}, \cdots, v_n a_{\pi(n)}\big).$$

To use R-SDP, we will make use of the following choices. First, we set

$$\mathbb{E} = \left\{ g^j \mid j \in \{0, 1, \cdots, z-1\} \right\},$$

where $g \in \mathbb{F}_q^*$ has multiplicative order $z < q - 1$. In other words, we choose $\mathbb{E}$ as the cyclic subgroup of $\mathbb{F}_q^*$ which is generated by g and, to have $\mathbb{E} \neq \mathbb{F}_q^*$, we require that g is not primitive. Then, we set $X := \mathscr{S}_w^{\mathbb{E}}$, i.e., choose the secret key as an element of the restricted Hamming sphere with radius w. Also, we set $\mathscr{T} := S_n \times \mathbb{E}^n$, which contains only the monomial transformations having restricted scaling coefficients. It is easy to see that, with these choices, (2) holds. Notice that the action of any $\sigma := (\pi, \mathbf{v}) \in \mathscr{T}$ can also be described as

$$\sigma(\mathbf{a}) = \pi(\mathbf{a}) \begin{pmatrix} v_1 & & \\ & \ddots & \\ & & v_n \end{pmatrix} = \pi(\mathbf{a}) \begin{pmatrix} g^{i_1} & & \\ & \ddots & \\ & & g^{i_n} \end{pmatrix}, \tag{3}$$

with $(i_1, \cdots, i_n) \in \mathbb{Z}_z^n$. We will refer to the set of all diagonal matrices as in (3) as the *restricted diagonal group*, which we denote by $D_n(g)$ and we refer to $\mathscr{T}$ as the group of *restricted isometries.* Note that this is a slight abuse of notation, since an isometry is a weight preserving map while, in our case, we use the term for the preservation of the restriction.

We observe that, when $w = n$, i.e., we have full weight, we can choose a simpler description for $\mathscr{T}$. Indeed, we have $\mathscr{S}_w^{\mathbb{E}} = \mathbb{E}^n$, so (2) holds even when $\mathscr{T} := D_n(g)$. Also, it can be seen that $\mathscr{T}$ acts transitively and freely on $X := \mathbb{E}^n$, which means that for any $\mathbf{e}, \mathbf{e}' \in \mathbb{E}^n$, there exists a unique $\sigma \in \mathscr{T}$ such that $\mathbf{e}' = \sigma(\mathbf{e})$. More interesting properties about this setting can be found in Section 5. For the moment, it is enough to anticipate that this choice is the one which will yield the best performances for ZK schemes.

4.2 The case study of CVE with R-SDP

The CVE protocol [17] has been, historically, the first ZK-ID scheme based on non-binary SDP with low Hamming weight. Modern solutions, such as [26], are basically built on CVE. Hence, as a preparatory step, it makes sense to start by adapting this protocol to the R-SDP setting. This allows us to show that the most common techniques to build a ZK protocol in the SDP setting hold also for the R-SDP setting. The CVE based on R-SDP is depicted in Figure 2.

It is easy to see that, as the original CVE scheme, the protocol achieves ZK. Indeed, $\mathbf{u}$ is chosen uniformly at random in $\mathbb{F}_q^n$ and the same holds for $\mathbf{e}' = \sigma(\mathbf{e})$, thanks to (2). Also, the soundness error remains the same as CVE, that is, $\varepsilon = \frac{q}{2(q-1)}$, and an adversary achieving a larger success probability is either able to solve R-SDP or find hash collisions. A rigorous proof of this fact would be identical to the one in [17] and is hence omitted.

For the considered R-SDP setting, we take into account two possible choices:

Private Key $\mathbf{e} \in \mathcal{S}_w^{\mathbb{E}}$
Public Key $\mathbb{E}, w, \mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}, \mathbf{s} = \mathbf{e}\mathbf{H}^\top \in \mathbb{F}_q^{n-k}$

PROVER		VERIFIER
Choose $\mathbf{u} \xleftarrow{\$} \mathbb{F}_q^n$, $\sigma \xleftarrow{\$} \mathscr{T}$		
Set $c_0 = \mathsf{Hash}\big(\sigma, \mathbf{u}\mathbf{H}^\top\big)$		
Set $c_1 = \mathsf{Hash}\big(\sigma(\mathbf{u}), \sigma(\mathbf{e})\big)$	$\xrightarrow{(c_0,c_1)}$	
	$\xleftarrow{\beta}$	Choose $\beta \xleftarrow{\$} \mathbb{F}_q^*$
Set $\mathbf{y} = \sigma(\mathbf{u} + \beta\mathbf{e})$		
	$\xrightarrow{\mathbf{y}}$	
	$\xleftarrow{b}$	Choose $b \xleftarrow{\$} \{0,1\}$
If $b = 0$, set $f := \sigma$		
If $b = 1$, set $f := \mathbf{e}' = \sigma(\mathbf{e})$	$\xrightarrow{f}$	If $b = 0$, accept if: $c_0 = \mathsf{Hash}\big(\sigma, \sigma^{-1}(\mathbf{y})\mathbf{H}^\top - \beta\mathbf{s}\big)$ If $b = 1$, accept if: $\mathbf{e}' \in \mathcal{S}_w^{\mathbb{E}}$ and $c_1 = \mathsf{Hash}\big(\mathbf{y} - \beta\mathbf{e}', \mathbf{e}'\big)$

Fig. 2: R-CVE: CVE scheme based on R-SDP

- *Choice* I: we consider moderate values of z such that from (1), we get $W^* < 1$. We set $w = W^* n < n$ and $\mathscr{T} := S_n \times \mathbb{E}^n$. Representing σ and $\mathbf{e}'$ requires

$$\begin{cases} \mathrm{Size}(\sigma) = n\lceil \log_2(n) \rceil + n\lceil \log_2(z) \rceil, \\ \mathrm{Size}(\mathbf{e}') = \min\{w, n-w\} \cdot \lceil \log_2(n) \rceil + w\lceil \log_2(z) \rceil. \end{cases} \tag{Choice I}$$

The expression for Size($\mathbf{e}'$) is derived considering that one can either specify the position of set entries (which requires $w\lceil \log_2(n) \rceil$ bits) or the positions of zeros (which requires $(n-w)\lceil \log_2(n) \rceil$ bits).

- *Choice* II: we consider values z for which (1) returns $W^* = 1$. Remember that, asymptotically, this is guaranteed when $z \approx q^{1-R}$. In this case, we can choose $\mathscr{T} := \mathbb{E}^n$, and consequently have

$$\mathrm{Size}(\sigma) = \mathrm{Size}(\mathbf{e}') = n\lceil \log_2(z) \rceil. \tag{Choice II}$$

When SDP is used, we instead have

$$\begin{cases} \mathrm{Size}(\sigma) = n\lceil \log_2(n) \rceil + n\lceil \log_2(q-1) \rceil, \\ \mathrm{Size}(\mathbf{e}') = \min\{w, n-w\} \cdot \lceil \log_2(n) \rceil + w\lceil \log_2(q-1) \rceil. \end{cases} \tag{SDP}$$

In Table 1 we have compared how the above sizes behave, when targeting a security level of $\lambda = 128$ bits. For SDP we have used the parameters which are recommended in [26], while for R-SDP we have designed some instances taking into account the attacks described in Appendix A. As we can see from the table, R-SDP outperforms SDP in all the considered quantities. In particular, Choice II seems to be better suited for ZK-ID schemes. Indeed, despite (slightly) larger values for n and q, representing σ becomes much easier, because we can get rid of the permutation part, which grows more than linearly in n.

Table 1: Comparison between communication costs for SDP and R-SDP, for the case of $\lambda = 128$ and $R \approx \frac{1}{2}$; all sizes are expressed in bytes.

	q	z	n	k	w	ISD cost	Size(σ)	Size($\mathbf{e}'$)	Size($\mathbf{y}$)
SDP	512	511	196	92	84	128	416.5	168.5	220.0
	256	255	207	93	90	128	414.0	168.8	207.0
	128	127	220	101	90	128	412.5	157.5	192.5
R-SDP I	677	26	84	42	73	128.1	126.0	55.3	105.0
	379	21	103	52	82	129.2	151.5	67.6	108.1
	197	14	103	51	91	128.9	141.6	56.0	103.0
R-SDP II	2017	63	70	32	70	130.1	52.5	52.5	96.3
	1021	30	79	40	79	130.0	49.4	49.4	98.8
	197	14	102	51	102	128.1	51.0	51.0	102.0

We would like to point out that, in modern protocols, several techniques can be applied to reduce the communication cost, the simplest one being sending generating seeds instead of random objects. However, almost every scheme (apart from those based on MPCItH) makes use, at some point, of messages containing the objects we have considered in Table 1. Seeing that the size to represent these quantities gets significantly reduced, when switching to R-SDP, is promising and gives motivation to consider how R-SDP can be applied also to modern protocols. The rest of this paper is dedicated to this task. From now on we will focus on Choice II, i.e., we will consider R-SDP with maximum Hamming weight $w = n$, since it allows for greater reductions. Additionally, this choice allows for much more significant improvements, described in Section 5.

4.3 (A sketch of) MPCItH protocols based on R-SDP

As another popular way to design a ZK protocol, one can use the MPCItH approach. For SDP, the MPCItH paradigm has been first employed in the SDItH scheme [24]. In the following, we just sketch the general idea behind SDItH and, for the sake of simplicity, omit several technical details.

The SDItH protocol simulates the interaction between N parties: the secret $\mathbf{e}$ gets split into N additive shares $\{\mathbf{e}^{(i)}\}_{1 \leq i \leq N}$ such that $\sum_{i=1}^{N} \mathbf{e}^{(i)} = \mathbf{e}$. The i-th user receives $\mathbf{e}^{(i)}$ and uses it to compute a share of the public key as $\mathbf{s}^{(i)} = \mathbf{e}^{(i)}\mathbf{H}^\top$. Then, the parties run a MPC protocol to verify that the sum of the shares $\mathbf{s}^{(i)}$ indeed corresponds to $\mathbf{s}$, and the shares $\mathbf{e}^{(i)}$ sum to a vector having the desired Hamming weight. For the latter, the authors of [24] propose a MPC protocol which is based on polynomial relations. The idea is that of constructing a polynomial whose degree is the same as the weight of $\mathbf{e}$: the degree verification demands for an ad-hoc MPC protocol.

For the case of R-SDP, using just the degree verification will not be enough. Indeed, the MPC parties should also verify that $\mathbf{e}$ has only restricted entries. So, converting the SDItH protocol to the use of R-SDP seems inappropriate.

However, the MPCItH protocol employed in [22, Section 6] for PKP should better fit our scopes. Indeed, the author proposes an MPC protocol to verify that two polynomials

$$P(x) = \prod_{j=1}^{n}(x - e_j), \quad P'(x) = \prod_{j=1}^{n}(x - e'_j), \tag{4}$$

for which the parties receive additive shares, have the same roots. This is used to check that two vectors $\mathbf{e}$ and $\mathbf{e}'$ are identical, up to a permutation. We now briefly sketch how a similar idea may be used also for R-SDP with maximum Hamming weight. Consider that the parties, from the shares $\mathbf{e}^{(i)}$, compute shares for the polynomial $P(x)$ as in (4). In our case, all the roots of $P(x)$ live in $\mathbb{E}$: this is what we can demand the MPC protocol to verify. Since the protocol also checks that $P(0) \neq 0$, this will convince the parties that $\mathbf{e}$ does not have zero entries and, consequently, has maximum Hamming weight.

We envision that a scheme using the above approach can be constructed with a protocol similar to the one in [22, Section 6]. However, we will not enter further details for two reasons. Firstly, since the isometries for R-SDP have a very small size, we believe the problem is more suited for protocols that employ isometries. Secondly, the BG scheme introduced in [14, Figure 3], which does not use the MPCItH paradigm, achieves smaller signature sizes than the PKP scheme described in [22, Section 6]. Since PKP is similar to a decoding problem, it is not surprising that also the BG scheme can be adapted to the R-SDP setting. We postpone the presentation of the resulting protocol to Section 6, and continue describing how R-SDP can be made even more powerful with an ad-hoc choice for the set of restricted vectors.

5 Using subgroups of the restricted diagonal group

In this section we show that, when the R-SDP with full Hamming weight is considered, a more compact representation for restricted objects can be obtained. The idea consists in identifying a set of restricted isometries that i) has small cardinality (but not too small, since this may facilitate attacks), and ii) admits a compact representation, which is preferably fast to compute. We extend this reasoning to restricted vectors, and in the end obtain that, for a security level of λ bits, we can represent any restricted object with $(1+\alpha)\lambda$ bits, with α being a small positive constant. This requires to introduce an additional security assumption: since we are reducing the space from which secret keys and ephemeral objects are sampled, security issues may arise. Yet, with coding theory arguments, we argue that incorporating this information into existing attacks does not lead to significant speed-ups.

5.1 Subgroups of the restricted diagonal group

Recall that $D_n(g) \subseteq \mathbb{F}_q^{n\times n}$ contains the matrices $\mathrm{diag}(g^{i_1}, \dots, g^{i_n})$, i.e., diagonal matrix whose diagonal is given by $(g^{i_1}, \dots, g^{i_n})$, with $i_j \in \{0, \dots, z-1\}$. Let us introduce the following bijection $\ell : D_n(g) \to \mathbb{Z}_z^n$, which allows for a vector representation of the matrices in $D_n(g)$, as

$$\ell\big(\mathrm{diag}(g^{i_1}, \dots, g^{i_n})\big) = (i_1, \dots, i_n).$$

It is easy to see that $(D_n(g), \cdot)$ is isomorphic to $(\mathbb{E}^n, \star)$, where " $\cdot$ " denotes the standard matrix multiplication and " $\star$ " denotes the component-wise multiplication. Additionally, both are abelian groups and ℓ is a group isomorphism to $(\mathbb{Z}_z^n, +)$. More generally, any $\mathbf{A}$ generates a cyclic subgroup $\left\{\mathbf{A}^i \mid i \in \mathbb{N}\right\} \subset D_n(g)$. Due to the isomorphism to $\mathbb{Z}_z^n$, the order of $\mathbf{A}$ is the same as the order of $\ell(\mathbf{A})$ in $(\mathbb{Z}_z^n, +)$. Recall that $x \in \mathbb{Z}_z$ has order $\frac{z}{\gcd(x,z)}$. Thus, for $\mathbf{A} = \mathrm{diag}(g^{i_1}, \dots, g^{i_n})$, we have that

$$\mathrm{ord}(\mathbf{A}) = \mathrm{lcm}\big(\mathrm{ord}(i_1), \dots, \mathrm{ord}(i_n)\big) = \mathrm{lcm}\left(\tfrac{z}{\gcd(i_1,z)}, \dots, \tfrac{z}{\gcd(i_n,z)}\right),$$

where lcm denotes the least common multiple.

Remark 2. One can easily construct a matrix $\mathbf{A}$ with maximum order z, by taking one of the i_j which is coprime to z.

We now consider the subgroup of $D_n(g)$ whose generating set is a set of m matrices from $D_n(g)$. Namely, we choose m matrices $\mathbf{B}_1, \dots, \mathbf{B}_m \in D_n(g)$, and define

$$G = \langle \mathbf{B}_1, \cdots, \mathbf{B}_m \rangle = \left\{ \prod_{j=1}^{m} \mathbf{B}_j^{u_j} \,\middle|\, u_i \in \{0, \dots, z-1\} \right\}.$$

In the following, we will call G the *restricted diagonal subgroup.* To any $\mathbf{A} \in G$, we can associate a vector representation through $\ell_G : G \to \mathbb{Z}_z^m$, as follows

$$\ell_G\left(\prod_{j=1}^{m} \mathbf{B}_j^{u_j}\right) = (u_1, \dots, u_m). \tag{5}$$

Clearly, $(G, \cdot) \subset (D_n(g), \cdot)$ is a subgroup and ℓ_G is a group homomorphism. Thus, for any $\mathbf{A} \in G, x \in \mathbb{N}$ we have $\ell_G(\mathbf{A}^x) = x\ell_G(\mathbf{A}) \mod z$.

Proposition 1. *Let $\mathbf{M}_G = \in \mathbb{Z}_z^{m\times n}$ be the matrix whose j-th row is $\ell(\mathbf{B}_j)$, and $\mathscr{B} = \{\mathbf{u}\mathbf{M}_G \mid \mathbf{u} \in \mathbb{Z}_z^m\}$. Then,*

1. *$\ell(\mathbf{A}) = \ell_G(\mathbf{A})\mathbf{M}_G \mod z$, for any $\mathbf{A} \in G$,*
2. *$|\mathscr{B}| = |G|$.*

Proof. Let $\mathbf{B}_j = \mathrm{diag}\left(g^{i_1^{(j)}}, \dots, g^{i_n^{(j)}}\right)$, hence $\ell(\mathbf{B}_j) = \left(i_1^{(j)}, \dots, i_n^{(j)}\right)$, and $\mathbf{A} = \prod_{j=1}^{m} \mathbf{B}_j^{u_j} \in G$. Then, it holds that

$$\mathbf{A} = \prod_{j=1}^{m} \mathrm{diag}\left(g^{u_j i_1^{(j)}}, \dots, g^{u_j i_n^{(j)}}\right) = \mathrm{diag}\left(g^{\sum_{j=1}^{m} u_j i_1^{(j)}}, \dots, g^{\sum_{j=1}^{m} u_j i_n^{(j)}}\right).$$

By construction, the element in the j-th row and v-th column of $\mathbf{M}_G$ is $i_v^{(j)}$. Hence, for $\mathbf{u} = \ell_G(\mathbf{A}) = (u_1, \ldots, u_m) \in \mathbb{Z}_z^m$ we get

$$\ell(\mathbf{A}) = \left(\sum_{j=1}^{m} u_j i_1^{(j)}, \ldots, \sum_{j=1}^{m} u_j i_n^{(j)} \right) = \mathbf{u}\mathbf{M}_G \in \mathbb{Z}_z^n.$$

The second claim follows, since $\ell : D_n(g) \mapsto \mathbb{Z}_z^n$ is a bijection. □

Example 1 Let $q = 13$ and $g = 5$, with multiplicative order $z = 4$; consequently

$$\mathbb{E} = \left\{ 1 = g^0,\ 5 = g^1,\ 12 = g^2,\ 8 = g^3 \right\}.$$

Let us consider $n = 5$ and $m = 3$. As generating set for G, we take

$$\mathbf{B}_1 = \mathrm{diag}(12, 5, 5, 5, 12),\ \ \mathbf{B}_2 = \mathrm{diag}(12, 1, 5, 5, 1),\ \ \mathbf{B}_3 = \mathrm{diag}(8, 12, 1, 1, 1)$$
$$\text{with } \ell(\mathbf{B}_1) = (2, 1, 1, 1, 2),\ \ \ell(\mathbf{B}_2) = (2, 0, 1, 1, 0),\ \ \ell(\mathbf{B}_3) = (3, 2, 0, 0, 0).$$

Each of these matrices has maximum order z and one can check that $|G| = |\mathscr{B}|$ is maximal, i.e., $z^m = 4^3 = 64$. Each matrix in G is associated with a length-3 vector over $\mathbb{Z}_4$. For instance, to $(1, 3, 0)$ we associate the matrix

$$\begin{aligned}
\mathbf{A} &= \ell_G^{-1}\big((1, 3, 0)\big) = \mathbf{B}_1^1 \mathbf{B}_2^3 \mathbf{B}_2^0 \\
&= \mathrm{diag}\big(g^2, g^1, g^1, g^1, g^2\big) \cdot \mathrm{diag}\big(g^2, g^0, g^3, g^3, g^0\big) \cdot \mathrm{diag}\big(g^0, g^0, g^0, g^0, g^0\big) \\
&= \mathrm{diag}\big(g^0, g^1, g^0, g^0, g^2\big) = \mathrm{diag}\big(1, 5, 1, 12\big).
\end{aligned}$$

We now make another example, considering $m = 2$; we analyze the group which is generated by the following matrices

$$\mathbf{B}_1 = \mathrm{diag}\big(1, 8, 8, 5, 1\big),\ \ \mathbf{B}_2 = \mathrm{diag}\big(1, 5, 5, 8, 12\big),$$

for which $\ell(\mathbf{B}_1) = (0, 3, 3, 1, 0)$ and $\ell(\mathbf{B}_2) = (0, 1, 1, 3, 2)$. It is easy to see that both $\mathbf{B}_1$ and $\mathbf{B}_2$ have multiplicative order z, but the order of G is not maximal, i.e., equal to $4^2 = 16$. Indeed, there exist linear combinations of the rows of $\mathbf{M}_G$ yielding the same vector: for any $(u_1, u_2) \in \mathbb{Z}_4^2$, it holds that $(2 + u_1)\ell(\mathbf{B}_1) + (2 + u_2)\ell(\mathbf{B}_2) \equiv u_1\ell(\mathbf{B}_1) + u_2\ell(\mathbf{B}_2) \mod 4$. Enumerating all the elements in $\mathscr{B}$, we find

$$\begin{aligned}
\mathscr{B} = \{&(0, 0, 0, 0, 0), (0, 3, 3, 1, 0), (0, 2, 2, 2, 0), (0, 1, 1, 3, 0), \\
&(0, 1, 1, 3, 2), (0, 0, 0, 0, 2), (0, 3, 3, 1, 2), (0, 2, 2, 2, 2)\}.
\end{aligned}$$

Consequently, $|G| = |\mathscr{B}| = 8$.

Remark 3. When z is a prime number, we can easily construct a G of maximal order z^m, by taking $\mathbf{B}_1, \ldots, \mathbf{B}_m \in D_n(g)$ such that $\{\ell(\mathbf{B}_1), \ldots, \ell(\mathbf{B}_m)\}$ are linearly independent. This is equivalent to asking for a full rank matrix $\mathbf{M}_G$.

Example 2 Let $q = 13$, $n = 5$, $m = 4$ and $g = 3$, with multiplicative order 3. We consider the group G whose generating set contains the matrices with the following vector representations $\ell(\mathbf{B}_i)$

$$(2, 0, 2, 0, 2),\ \ (2, 2, 0, 2, 2),\ \ (0, 2, 2, 1, 1),\ \ (1, 2, 2, 2, 2).$$

The resulting $\mathbf{M}_G$ has full rank, i.e., 4, so $\mathscr{B}$ and G contain $3^4 = 81$ elements.

5.2 Solving R-SDP with restricted diagonal subgroup

From now on, we focus only on restrictions $\mathbb{E} = \{g^i \mid i \in \{0, \cdots, z-1\}\}$ such that z is prime. Also, we consider only restricted diagonal subgroups G having maximum order $|G| = z^m$. We now consider R-SDP with the additional constraint that the solution must be associated with an element of G; the corresponding problem is defined as follows.

Problem 3 R-SDP(G): SDP with Restricted Diagonal Subgroup G
Let $G = \langle \mathbf{B}_1, \ldots, \mathbf{B}_m \rangle$, $\mathbf{H} \in \mathbb{F}_q^{r\times n}$ *and* $\mathbf{s} \in \mathbb{F}_q^r$. *Does there exist a vector* $\mathbf{e}$ *such that* $\mathrm{diag}(\mathbf{e}) \in G$ *and* $\mathbf{e}\mathbf{H}^\top = \mathbf{s}$*?*

For the sake of simplicity, we will sometimes slightly abuse notation and will say $\mathbf{e} \in G$, (obviously implying that $\mathrm{diag}(\mathbf{e}) \in G$). When $G = D_n(g)$, i.e., R-SDP(G) corresponds to R-SDP.

Notice that R-SDP(G) admits less solutions than the more general R-SDP. Consequently, we can modify the criterion to have a unique solution as

$$|G|q^{-(1-R)n} < 1$$

and since $|G| = z^m$ we get $m\log_2(z) - (1-R)n\log_2(q) \leq 0$.

At a first glance, it seems like R-SDP(G) cannot be harder than R-SDP. Indeed, the solution space now is G, with size $|G| \leq z^m$, instead of $\mathbb{E}^n$, which is bigger and has size z^n. So, there may be attacks that exploit this additional constraint. Consequently, the following questions naturally arise: *Under which conditions is the R-SDP(G) easier than R-SDP?* And: *How can we exploit the knowledge of G?*

It is obvious that, when G is chosen improperly, the problem may become much easier than R-SDP. For instance, when G has a very small order, the R-SDP(G) can be solved with a trivial brute force attack over G, taking time $O\big(|G|\big)$. So, it is important that G has a sufficiently large order, say, $|G| \geq 2^\lambda$.

On the other hand, one can also disregard G, enumerate all solutions and check their validity afterwards. However, such attacks can be easily thwarted by choosing instances which have more than 2^λ solutions, when G is neglected. In this case, already the cost of checking the validity guarantees the security level. It follows that any efficient solver for R-SDP(G) has to directly take G into account. Since this is not possible for the algorithm presented in Appendix A, we only consider restricted Stern in the following. An improved collision search requires a method to enumerate parts of the solution vector with size $t \geq k/2$ in time smaller than z^t. This can be done with the following procedure, which starts from a set $J \subseteq \{1, \ldots, n\}$ of size t and returns all candidates for $\mathbf{e}_J$:

1) set $\mathbf{M}'$ as the $m \times t$ sub-matrix formed by the columns of $\mathbf{M}_G$ which are indexed by J;
2) enumerate all length-t vectors which can be obtained as linear combinations of the rows of $\mathbf{M}'$;

3) use any such vector as a series of exponents, and generate the corresponding candidate for $\mathbf{e}_J$. To do this, one first enumerates

$$\mathscr{B}' = \{\mathbf{u}\mathbf{M}' \mid \mathbf{u} \in \mathbb{F}_z^m\} \subseteq \mathbb{F}_z^t.$$

Then, to each $\mathbf{x} \in \mathscr{B}'$, associates a candidate for $\mathbf{e}_J$ as $\ell^{-1}(\mathbf{x})$, i.e., uses the elements in $\mathbf{x}$ as the exponents for g.

With the above approach, one can enumerate all candidates for $\mathbf{e}_J$ in time $O(|\mathscr{B}'|) = O\big(z^{m'}\big)$, where $m' = \mathrm{rank}(\mathbf{M}')$. Indeed, it is enough to find a basis for $\mathscr{B}'$, of size $m' \times t$, and then enumerate all the vectors which can be generated as linear combinations of the basis elements. Notice that $m' \leq \min\{m, t\}$. If $m' = t$, we have $\mathscr{B}' = \mathbb{F}_z^t$, so that enumeration takes time $O(z^t)$, that is, the same that one would face without taking into account the structure of G. Consequently, one can hope for some kind of improvement only if $m' < t$.

Example 3 Let us consider the case of $q = 11$ and $g = 3$, having order $z = 5$. Let $n = 10$ and $m = 3$, and assume that the considered group G is such that

$$\mathbf{M}_G = \begin{pmatrix} 1\,4\,3\,4\,2\,3\,2\,0\,1\,0 \\ 0\,2\,1\,0\,1\,2\,4\,2\,3\,3 \\ 1\,3\,3\,4\,3\,0\,1\,4\,4\,3 \end{pmatrix},$$

with rank 3. Hence, the group G has maximum order 5^3. Let $t = 4$ and consider $J = \{1, 2, 3, 4\}$. The columns of $\mathbf{M}_G$ which are indexed by J form a matrix $\mathbf{M}'$ with the three linearly independent rows. Consequently, $m' = \mathrm{rk}(\mathbf{M}') = 3$ and to enumerate all candidates for $\mathbf{e}_J$ it is enough to enumerate all the exponents vectors which can be generated by linear combinations of the rows of $\mathbf{M}'$. Instead of $z^t = 5^4$, we can enumerate all candidates for $\mathbf{e}_J$ using time $z^{m'} = 5^3$. However, if $J = \{4, 6, 7, 9\}$, then the corresponding columns form a matrix $\mathbf{M}'$ with rank 2. Thus, we can enumerate all candidates for $\mathbf{e}_J$ in time $z^{m'} = 5^2$.

The problem of finding a set J with the desired properties can be stated as follows.

Problem 4 (Submatrix Rank Problem) *Let $\mathbf{M}_G \in \mathbb{F}_z^{m \times n}$, with $m < n$ and $m' \leq m$. Is there a set $J \subset \{1, \ldots, n\}$ of size t, such that $rk((\mathbf{M}_G)_J) = m'$?*

Assuming that one is able to find a set J such that $\mathbf{M}' = (\mathbf{M}_G)_J$ has rank $m' < t$, one can possibly speed-up ISD algorithms:

- if $t > k$, then J contains an information set $J' \subseteq J$[6]. So, we are able to enumerate all candidates for $\mathbf{e}_{J'}$ in time $z^{m'}$. If m' is particularly low (say, lower than $\lambda \log_2(z)$) the attack can use a single list of size $z^{m'}$ in which we put candidates for $\mathbf{e}_{J'}$. See Figure 3a for a representation of this strategy;

[6] Unless all sets of k which are contained in J are not information sets. However, this happens with extremely low probability.

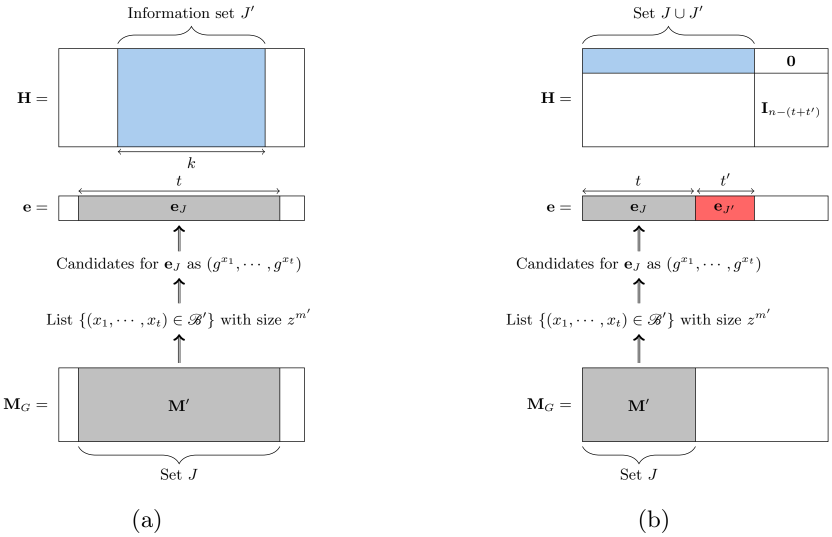

Fig. 3: Strategies to speed-up ISD with the knowledge about G.

- if $t < k$, then we can use the $z^{m'}$ candidates for $\mathbf{e}_J$ to build one of the lists for Stern's algorithm. However, we also require an enumeration of all $\mathbf{e}_{J'}$, with J' disjoint from J, of size t', such that $t + t' \geq k$. Thus, a collision search leads to a cost of
$$\max\left\{z^{m'}, z^{t'}, z^{m+m'}q^{k-(t+t')}\right\}.$$
So, this approach can be convenient only if $t \geq k/2$. See Figure 3b for a representation of this strategy;
- assume that one is able to find several sets J of size t such that $m' < t$. Then, we can enumerate several portions of the solution $\mathbf{e}$, of size t, in time $z^{m'}$. We can use them to build several lists, which we can combine with a collision search approach with more than one level. Again, there is no guarantee that this yields an attack with overall cost $z^{m'}$ since, in practice, we need to take into account how list sizes grow after merging.

To completely cut out all the above possibilities, we will adopt the following very conservative criterion.

Requirement 1 *We want $\mathbf{M}_G \in \mathbb{F}_z^{m \times n}$ such that, for any $t \in \{1, \ldots, n\}$ and any set $J \subseteq \{1, \ldots, n\}$ of size t, we have rk$((\mathbf{M}_G)_J) > \lambda \log_z(2)$.*

This implies that any strategy that exploits the structure of G to speed-up ISD attacks will not be more efficient than generic ISD attacks, which run in time not lower than 2^λ (since we are focusing on instances that achieve λ bits of security).

We now provide strong evidences that Requirement 1 is rather conservative. First, we show that Problem 4 is NP-hard. This implies that, even if some set J

with the desired properties exists, finding it cannot be too easy. The NP-hardness proof will make use of the following result.

Theorem 1. Relation between m' and subcodes of $\mathscr{B}$
Let $\mathbf{M}_G \in \mathbb{F}_z^{m \times n}$ and $\mathscr{B} = \langle \mathbf{M}_G \rangle \subseteq \mathbb{F}_z^n$ be a linear code of dimension m. Then, there exists $J \subseteq \{1, \dots, n\}$ of size t, such that $m' = rk((\mathbf{M}_G)_J)$, if and only if

i) if $m' \leq t \leq m$, then $\mathscr{B}^\perp$ contains a subcode with dimension $t - m'$ and support size $\leq t$;
ii) if $m' \leq m \leq t$, then $\mathscr{B}$ contains a subcode with dimension $m - m'$ and support size $\leq n - t$.

Proof. We start with the case $m' \leq t \leq m$. Since $\mathbf{M}' = (\mathbf{M}_G)_J$ has m rows and $t \leq m$ columns, if its rank is lower than t this implies that there exist $k' = t - m'$ linearly independent vectors $\mathbf{x}_1, \cdots, \mathbf{x}_{k'} \in \mathbb{F}_z^t$ such that $\mathbf{M}'\mathbf{x}_i^\top = \mathbf{0}$. We can use such vectors to define a basis $\mathbf{X} \in \mathbb{F}_z^{k' \times t}$ for the right kernel of $\mathbf{M}'$. Now, let $\mathbf{C} \in \mathbb{F}_q^{k' \times n}$ be a matrix such that $\mathbf{C}_J = \mathbf{X}$ and $\mathbf{C}_{J^C} = \mathbf{0} \in \mathbb{F}_z^{k' \times (n-t)}$. By construction, it holds that $\mathbf{C}$ has rank k' and is such that $\mathbf{M}_G\mathbf{C}^\top = \mathbf{0}$, so $\mathbf{C}$ is a generator for a k'-dimensional subcode of $\mathscr{B}^\perp$. Since $\mathbf{C}$ has at least $n - t$ null columns (the ones indexed by J^C), we know that $\mathbf{C}$ generates a code with dimension k' and support size not greater than t. For the other direction, one can proceed in the exact same way. If there is a subcode of $\mathscr{B}^\perp$ of dimension $t - m'$ and support size $\leq t$, we can find a generator matrix $\mathbf{C}$, which has (at least) $n - t$ zero columns and denote these indices by J^C.

The case of $m' \leq m \leq t$ is treated analogously, with the only difference that we need to focus on the left kernel of $\mathbf{M}'$. □

Theorem 2. *The Submatrix Rank Problem is NP-complete.*

The proof, which is shown in Appendix B, follows from a reduction from the problem of finding low weight codewords in a given code, which is one of the foundational hard problems in code-based cryptography.

Notice that, as a consequence of Theorem 1, finding sets J with the desired properties implies finding subcodes with small supports. This can be done using ISD, with a time complexity that (more or less) grows exponentially with the desired support size. Thus, finding a set J may also be unfeasible. However, we describe how to choose the value of m so that such useful subcodes are not expected to exist. For a random code with length n and dimension k, over $\mathbb{F}_z$, the average number of subcodes with dimension k' and support size w is well estimated by [31, Theorem 1]

$$N_k(k', w) = \binom{n}{w} (z^{k'} - 1)^{w-k'} \left[{k \atop k'}\right]_z \left[{n \atop k'}\right]_z^{-1}. \tag{6}$$

Since for $\mathbf{M}_G$ we do not impose any structure, apart from the full rank property, we can safely study its row space $\mathscr{B}$ as a random code with dimension m. Analogously, we can treat its dual $\mathscr{B}^\perp$ as a random code as well, with dimension $n - m$. So, we can update Requirement 1 as follows.

Requirement 2 *We set $m > \lambda\log_z(2)$ as the minimum integer such that*

- *for any $m' \leq t \leq m$ with $N_{n-m}(t-m', t) > 0$, we have $m' > \lambda\log_z(2)$;*
- *for any $m' \leq m < t$ with $N_m(m-m', n-t) > 0$, we have $m' > \lambda\log_z(2)$.*

6 Building ZK-ID schemes from the R-SDP: modern protocols

Let us describe how using R-SDP and R-SDP(G) for the design of ZK protocols can lead to significant reductions in the signature size. First, we briefly comment on how R-SDP(G) can be implemented, and then we describe how the GPS [26] and BG-PKP protocols [14] can be converted to use R-SDP and R-SDP(G).

6.1 R-SDP(G) in practice: easy to implement and tight parameters

When R-SDP(G) is used, the generating set $\langle \mathbf{B}_1, \dots, \mathbf{B}_m \rangle$ must be publicly known. It can be easily made part of the public key: the prover samples a seed $\mathtt{Seed}_G$ to generate a candidate for $\langle \mathbf{B}_1, \dots, \mathbf{B}_m \rangle$ and checks if the corresponding $\mathbf{M}_G$ has maximum rank m. If this is not true, one discards the seed and restarts. When a valid seed is found, the prover samples $\mathbf{e}$ at random from G and computes $\mathbf{s} = \mathbf{e}\mathbf{H}^\top$, where $\mathbf{H}$ can be sampled from the seed: the secret key will be $\mathbf{e}$, the public key is $\{\mathbf{s}, \mathtt{Seed}\}$ of size $(n-k)\log_2(q) + |\mathtt{Seed}|$. We will generically set $|\mathtt{Seed}| = \lambda$. Notice that, as an option, one can even fix G, i.e., fix a generating set $\langle \mathbf{G}_1, \dots, \mathbf{G}_m \rangle$ and use it for every instantiation of the protocol.

To sample uniformly at random some $\mathbf{A} \in G$, one can first sample $\mathbf{u} \xleftarrow{\$} \mathbb{F}_z^m$ and then compute $\mathbf{A} = \ell_G^{-1}(\mathbf{u}) = \prod_{j=1}^m \mathbf{B}_j^{u_j}$. In practice, this can be done by first computing $(v_1, \dots, v_n) = \mathbf{u}\mathbf{M}_G$, which requires $O(nm)$ operations[7] over $\mathbb{F}_z$, and then using the indices to generate the restricted entries as $\big(g^{v_1}, \dots, g^{v_m}\big)$, which requires $O(n)$ operations over $\mathbb{F}_q$. Then, the obtained n values will either be considered as the elements of a restricted vector, or as the diagonal defining a restricted isometry. As it is common in ZK protocols, random objects will be communicated using the generating seed, of size λ, with which a secure PRNG has been fed. When R-SDP(G) is employed, we consider that the PRNG outputs a vector $\mathbf{u} \in \mathbb{F}_z^m$, and then generates the associated restricted object as $\ell_G^{-1}(\mathbf{u})$.

To verify that a given $\mathbf{a}$ is indeed in G, it is enough to check that $\ell(\mathbf{a})$ is a linear combination of the rows of $\mathbf{M}_G$. This can be done using a basis $\mathbf{C} \in \mathbb{F}_z^{(n-m)\times n}$ for the null space of $\mathbf{M}_G$: $\mathbf{a} \in G$, if and only if $\ell(\mathbf{a})\mathbf{C}^\top = \mathbf{0}$.

We consider that both restricted isometries and vectors are always sampled from G. The validity of (2), which guarantees that a scheme achieves ZK, holds since G acts transitively and freely on itself. In other words, for any $\mathbf{e} \in G$, $\sigma(\mathbf{e})$ is uniformly random over G when $\sigma \xleftarrow{\$} G$. When $\mathbf{e}$ and $\widetilde{\mathbf{e}}$ are two restricted vectors,

[7] Notice that this number can be reduced if one considers a convenient representation for the generating set. For instance, $\mathbf{M}_G$ can have the $m \times m$ identity on the left: computing the exponents will take time $O\big(m(n-m)\big)$.

Table 2: Instances of R-SDP(G) for $\lambda = 128$ and corresponding sizes for objects expressed in bytes.

Range for q	$\frac{z}{q-1}$	q	z	n	k	m	α	Size(σ)	Size$(\mathbf{y})$
$2^8 < q < 2^{10}$	1/2	1019	509	40	16	18	0.2644	20.2	49.9
		347	173	41	20	23	0.3359	21.4	43.2
		719	359	49	17	20	0.3262	21.2	58.1
	< 1/2	971	97	44	26	26	0.3406	21.4	54.8
		643	107	60	25	26	0.3604	21.9	70.0
		269	67	52	27	29	0.3743	21.9	52.5
$2^6 < q < 2^8$	1/2	227	113	43	22	24	0.2789	20.5	42.1
		107	53	53	26	31	0.3872	22.2	44.7
		83	41	73	28	35	0.4650	23.4	58.2
	< 1/2	223	37	56	33	34	0.3838	22.1	54.6
		103	17	76	44	48	0.5328	24.5	63.5
		79	13	82	49	54	0.5611	25.0	64.6
$2^4 < q < 2^6$	1/2	59	29	63	31	38	0.4422	23.1	46.3
		47	23	69	34	42	0.4843	23.7	47.9
		23	11	93	46	61	0.6486	26.4	52.6
	< 1/2	53	13	82	47	54	0.5611	25.0	58.7

the isometry σ that maps $\widetilde{\mathbf{e}}$ into $\mathbf{e}$ is $\ell_G(\mathbf{e}) - \ell_G(\widetilde{\mathbf{e}})$ and it can be represented using only $m\log_2(z) = (1+\alpha)\lambda$ bits.

We show that, even if we consider the conservative criteria of Requirement 2, we can use parameters that are much more aggressive than those we can have with R-SDP. From now on we will write $|G| = z^m = 2^{(1+\alpha)\lambda}$: the value of α gives an idea of how tight we can be, when representing elements of G. Obviously, $\alpha < 0$ is not a good idea, due to brute force attacks over G. Yet, if we can choose $\alpha < 1$, then we can have a size for restricted objects which is not greater than 2λ, that is, the binary size of a digest. In other words, we are making restricted objects smaller than some of the objects that the parties cannot avoid to exchange (e.g., the initial commitments). As we show in the following, in practice we can use values in the range $0.2 \div 0.6$: we are very close to do security with the minimum amount of required bits (that is, λ bits).

Some example instances for R-SDP(G) are shown in Table 2. We see that there are several trade-offs in how parameters can be chosen. For instance, large values of q lead to slightly smaller sizes for $\mathbf{y}$, while the arithmetic over $\mathbb{F}_q$ becomes slower. Another degree of freedom is in the choice of z: setting $z = \frac{q-1}{2}$ leads to smaller sizes, but values of z that are too large might need to be avoided for the sake of a faster arithmetic over $\mathbb{F}_z$. Comparing these numbers with those in Table 1, we see that using R-SDP(G) allows to reduce significantly the communication cost. In the next two sections we apply the problem to existing ZK schemes and derive their performances in terms of signature size.

6.2 R-GPS: the GPS scheme with R-SDP

The GPS scheme [26] applies the protocol-with-helper paradigm to the CVE scheme. In a nutshell, the idea is that of simulating a trusted third entity (the helper), which generates some of the messages which would be exchanged between the prover and the verifier. The helper is asked to generate the commitments and the first public response (that is, c_0, c_1 and $\mathbf{y}$ for the scheme in Figure 2). To remove the helper, the *cut&choose* technique is used. The helper is first simulated by the prover for N rounds, generating random objects from seeds and commiting to the obtained quantities. The verifier will ask to *open* only $M < N$ rounds: she will receive the verifying isometries for the chosen rounds, and the seeds to repeat the helper simulation for the other $N - M$ rounds.

Since GPS is based on SDP, converting it to R-SDP is rather straightforward and, due to lack of space, operational details about the resulting protocol are not reported in this paper. Also, the security analysis and signature size easily follows from [26]. To prevent the attack in [28], N and M must be such that

$$\max_{x\in\{0,\ldots,M\}} \binom{N-x}{M-x}\binom{N}{M}^{-1}(q-1)^{x-M} \geq 2^{\lambda}. \tag{7}$$

When R-SDP(G) is used, the communication cost of an opened round is

$$L = \underbrace{n\log_2(q)}_{\mathbf{y}\in\mathbb{F}_q^n} + \underbrace{2\lambda}_{\text{Randomness}} + \underbrace{m\log_2(z)}_{\sigma\in G}. \tag{8}$$

When relying on R-SDP, the resulting communication cost is obtained by replacing $m\log_2(z)$ with $n\log_2(z)$. The size of a signature in the resulting R-GPS signature scheme is

$$|\texttt{Signature}| = \underbrace{2\lambda\left(2 + M\log_2\left(\frac{N(q-1)}{M}\right)\right)}_{\text{Merkle proofs and commitments}} + \underbrace{\lambda M\log_2\left(\frac{N}{M}\right)}_{\text{Seeds}} + M\cdot L.$$

In Table 3 we report examples for the signature sizes we can achieve and compare them with the ones in [26, Table 1]. To have a fair comparison, we considered analogous values of N and M, so that reductions in the signature size are only due to the use of a different problem. As we expected, R-SDP always beats SDP, since signature sizes get reduced by a factor approximately 0.6. Considering R-SDP(G), the gain becomes more significant and, with respect to the instances based on R-SDP, we save approximately $1 \div 2$ kB.

6.3 R-BG: the BG-PKP scheme with R-SDP

As another protocol-with-helper, one may consider the FJR scheme [23]. To reduce the soundness error, FJR uses the idea of shared permutations: the random masking is obtained by combining the actions of N random permutations, so that a cheating prover cannot cheat for more than one permutation. This reduces the

Table 3: Performances of the GPS scheme [26] based on different problems.

	q	z	n	k	w	m	N	M	Sign. Size (kB)
SDP	128		220	101	90		512	23	24.6
	256		207	93	90		1024	19	22.4
	512		196	92	84		2024	16	20.6
	1024		187	90	80		4096	14	19.5
R-SDP	67	11	147	63	147		512	24	14.8
	197	14	105	53	105		1024	19	13.4
	991	33	77	48	77		2048	16	12.9
	991	33	77	38	77		4096	14	12.5
R-SDP(G)	53	13	82	47	82	54	512	25	12.7
	103	17	76	44	76	48	1024	21	12.7
	223	37	56	33	56	34	2048	19	11.8
	1019	509	40	16	40	18	4096	14	11.5

soundness error of a single round to $1/N$. We can adapt also this scheme to the R-SDP setting, but this will not lead to a great improvement, as before with respect to the GPS protocol. Nonetheless, the idea of shared permutations has been applied also to PKP, for a protocol that we will refer to as BG-PKP [14]. Notice that BG-PKP is the PKP based scheme with the smallest signatures. We show that, with minor modifications, the scheme can be adapted to the R-SDP setting and derive the resulting signature sizes.

For PKP, the prover first samples a vector $\mathbf{e} \in \mathbb{F}_q^n$, a full rank $\mathbf{H} \in \mathbb{F}_q^{r \times n}$, a permutation $\pi \in S_n$ and computes $\mathbf{s} = \pi(\mathbf{e})\mathbf{H}^\top$. The secret key is the permutation π and the public key is $\{\mathbf{H}, \mathbf{e}, \mathbf{s}\}$. For R-SDP, we can do the same. The only differences are that $\mathbf{e}$ and the transformation are sampled from G; namely, once $\mathbf{H}$ and G have been defined, we sample $\mathbf{e}, \sigma \xleftarrow{\$} G$, set the secret key as σ and the public key as $\{G,\ \mathbf{H},\ \mathbf{e},\ \mathbf{s} = \sigma(\mathbf{e})\mathbf{H}^\top\}$. Obviously, to compress the public key size, everything but $\mathbf{s}$ can be generated from a seed $\mathtt{Seed}^{(pk)}$.

The resulting protocol is shown in Figure 4. We have implicitly introduced some additional notation: $\mathsf{SeedTree}$, $\mathsf{SeedPath}$ are the functions to operate with the seeds tree (respectively, generate the tree from a master seed, compute a path and regenerate all seeds but one), while $\xleftarrow{\mathsf{Seed}}$ means sampling with randomness source $\mathtt{Seed}$. It can be seen that the protocol structure is the same as BG, so it inherits all of its features. As in [14, Theorem 2], the soundness error is

$$\varepsilon(N, q) = \frac{1}{N} + \frac{N-1}{N(q-1)}.$$

To obtain a signature scheme, we consider t parallel executions and then apply the Fiat-Shamir transform. To set the value of t so that the attack in [28] is

```
Private Key   $\sigma \in G$
Public Key    $G = \langle \mathbf{B}_1, \cdots, \mathbf{B}_m \rangle$, $\mathbf{e} \in G$, $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $\mathbf{s} = \sigma(\mathbf{e})\mathbf{H}^\top \in \mathbb{F}_q^{n-k}$
PROVER                                                                              VERIFIER
Sample $\mathtt{MSeed} \xleftarrow{\$} \{0;1\}^\lambda$
Compute $\{\mathtt{Seed}_i\}_{1\le i\le N} = \mathsf{SeedTree}(\mathtt{MSeed})$
For $i = 2, \cdots, N$:
  Sample $\mathtt{Seed}_i^*, \mathtt{Salt}_i \xleftarrow{\mathtt{Seed}_i} \{0;1\}^\lambda$
  Sample $\sigma_i \xleftarrow{\mathtt{Seed}_i^*} G$, $\mathbf{v}_i \xleftarrow{\mathtt{Seed}_i^*} \mathbb{F}_q^n$
  Set $c_i = \mathsf{Hash}\big(\mathtt{Salt}_i, \mathtt{Seed}_i^*\big)$
Set $\sigma_1 = \sigma_2^{-1} \circ \cdots \sigma_N^{-1} \circ \sigma$
Sample $\mathtt{Seed}_1^*, \mathtt{Salt}_1 \xleftarrow{\mathtt{Seed}_1} \{0;1\}^\lambda$
Sample $\mathbf{v}_1 \xleftarrow{\mathtt{Seed}_1^*} \mathbb{F}_q^n$
Set $c_1 = \mathsf{Hash}\big(\mathtt{Salt}_1, \mathtt{Seed}_1^*, \sigma_1\big)$
Compute $\mathbf{v} = \mathbf{v}_N + \sum_{i=1}^{N-1} \sigma_N \circ \cdots \circ \sigma_{i+1}(\mathbf{v}_i)$
Set $c = \mathsf{Hash}\big(\mathbf{v}\mathbf{H}^\top, \{c_i\}_{1\le i\le N}\big)$         $\xrightarrow{c}$
                                                                $\xleftarrow{\beta}$    Sample $\beta \xleftarrow{\$} \mathbb{F}_q^*$
Set $\widetilde{\mathbf{e}}_0 = \beta\mathbf{e}$
For $i = 1, \cdots, N$:
  Set $\widetilde{\mathbf{e}}_i = \sigma_i(\widetilde{\mathbf{e}}_{i-1}) + \mathbf{v}_i$
Set $h = \mathsf{Hash}\big(\{\widetilde{\mathbf{e}}_i\}_{1\le i\le N}\}\big)$         $\xrightarrow{h}$
                                                                $\xleftarrow{i}$    Sample $i \xleftarrow{\$} \{1, \cdots, N\}$
Compute $\mathtt{Seeds} = \mathsf{SeedPath}(\mathtt{MSeed}, i)$
If $i \neq 1$:
  Set $\mathtt{Resp} = \{c_i, \widetilde{\mathbf{e}}_i, \sigma_1, \mathtt{Seeds}\}$
Else:
  Set $\mathtt{Resp} = \{c_i, \widetilde{\mathbf{e}}_i, \mathtt{Seeds}\}$         $\xrightarrow{\mathtt{Resp}}$
                                                                If $i \neq 1$, verify $\sigma_1 \in G$
                                                                Generate $\{\mathtt{Seed}_j\}_{j\neq i} = \mathsf{GenSeeds}(\mathtt{Seeds})$
                                                                For $j \neq i$:
                                                                  Sample $\mathtt{Seed}_j^*, \mathtt{Salt}_j \xleftarrow{\mathtt{Seed}_j} \{0;1\}^\lambda$
                                                                  Sample $\sigma_j \xleftarrow{\mathtt{Seed}_j^*} G$, $\mathbf{v}_j \xleftarrow{\mathtt{Seed}_j^*} \mathbb{F}_q^n$
                                                                  Set $c_j = \mathsf{Hash}\big(\mathtt{Salt}_j, \mathtt{Seed}_j^*\big)$
                                                                Set $\widetilde{\mathbf{e}}_0 = \beta\mathbf{e}$
                                                                For $j \neq i$:
                                                                  $\widetilde{\mathbf{e}}_j = \sigma_j(\widetilde{\mathbf{e}}_{j-1}) + \mathbf{v}_j$
                                                                  If $j \neq 1$:
                                                                    Compute $c_j = \mathsf{Hash}\big(\mathtt{Salt}_j, \mathtt{Seed}_j^*\big)$
                                                                  Else:
                                                                    Compute $c_1 = \mathsf{Hash}\big(\mathtt{Salt}_1, \mathtt{Seed}_1^*, \sigma_1\big)$
                                                                Compute $\widetilde{\mathbf{s}} = \widetilde{\mathbf{e}}_N\mathbf{H}^\top - \beta\mathbf{s}$
                                                                Verify $c = \mathsf{Hash}\big(\widetilde{\mathbf{s}}, \{c_j\}_{1\le j\le N}\big)$
                                                                Verify $h = \mathsf{Hash}\big(\{\widetilde{\mathbf{e}}_j\}_{1\le j\le N}\big)$
```

Fig. 4: One round of the R-BG protocol

mitigated, we rely on the analysis in [14, Section 4.2]. To this end, let

$$P(t', t, N) = \sum_{j=t'}^{t} \binom{t}{j} \left(\frac{1}{q-1}\right)^j \left(\frac{N-1}{N}\right)^{t-j},$$

$$t^* = \arg\min_{0\le x\le t} \left\{ \frac{1}{P(x,t,N)} + N^{t-x} \right\}.$$

Table 4: Performances of the BG-PKP scheme [14] based on different problems

	q	z	n	k	m	N	t	Sign. Size (kB)
PKP	997		61	33		32	42	10.0
						256	31	8.9
R-SDP	991	33	77	38		32	42	10.8
						256	31	9.5
R-SDP(G)	971	97	44	26	26	32	42	8.0
						256	31	7.4
	1019	509	40	16	18	32	42	7.7
						256	31	7.2

Then, we choose t so that $P(t^*, t, N)^{-1} + N^{t-t^*} > 2^\lambda$. The signature size can easily be estimated as

$$|\texttt{Signature}| = 5\lambda + t\big(\underbrace{n\log_2(q)}_{\widetilde{\mathbf{e}}_i} + \underbrace{m\log_2(z)}_{\sigma_1} + \underbrace{\lambda\log_2(N)}_{\text{Seeds}} + \underbrace{2\lambda}_{c_i}\big).$$

When R-SDP with maximum weight is used, $n\log_2(z)$ gets replaced by $m\log_2(z)$.

Some instances of the resulting signature scheme are reported in Table 4. Signatures obtained from R-SDP are slightly larger than those based on PKP; instead, when using R-SDP(G), we achieve significant reductions with respect to R-SDP and, ultimately, beat PKP. We have considered both the case of $z = (q-1)/2$ and $z \ll q$; the one with smaller z has slightly larger signatures but, when implemented, should lead to a faster scheme, since arithmetic over a smaller $\mathbb{F}_z$ is faster.

6.4 Comparison with other post-quantum signatures

In Table 5 we compare the proposed schemes with other post-quantum signatures. As it is common in the literature, we have distinguished between "fast" variants (those with the lowest number of rounds, that is, with a smaller computational cost) and "short" variants (the ones with a larger number of rounds and shorter signatures). Our protocols compare very favourably with the schemes existing in the literature, even when considering the more conservative R-SDP.

For what concerns SDP, we achieve signatures that are smaller than those of all other schemes, apart from some variants of the Ret. of SDitH and WAVE. Notice that WAVE is a Hash&Sign scheme and has large public keys (more than 3MB); our protocols, instead, use public keys of less than 0.1 kB.

Schemes based on the rank metric can achieve smaller signatures when some structure is considered (e.g., ideal codes). An exception is Durandal, which is not obtained from a ZK protocol, but has much larger public keys. An analogous situation holds for LESS-FM. Finally, our R-BG protocol beats all existing schemes based on PKP, and has signatures that are smaller than both variants of SPHINCS$^+$.

Table 5: Comparison between signature schemes based on different post-quantum problems for $\lambda = 128$; all sizes are expressed in kB.

Problem	**Scheme**	**Pk size**	**Sign. size**	**Pk+Sign. size**	**Variant**
Hamming SDP, low weight	GPS[26]	0.1	24.0	24.1	Fast
		0.1	19.8	19.9	Short
	FJR[23]	0.1	22.6	22.7	Fast
		0.1	16.0	16.1	Short
	SDItH[24]	0.1	11.5	11.6	Fast
		0.1	8.3	8.4	Short
	Ret. of SDitH[2]	0.1	12.1	12.1	Fast, Var. 3
		0.1	5.7	5.8	Shortest, Var.3
Hamming SDP, large weight	WAVE[18]	3200	2.1	3202	-
Code Equivalence	LESS-FM[6]	10.4	11.6	23.0	Balanced
		205.7	5.3	211.0	Short sign
Rank Syndrome Decoding	Fen[22]	0.1	11.0	11.1	Fast
		0.1	8.5	8.6	Short
	BG[14]	0.1	17.2	17.3	Fast
		0.1	12.6	12.7	Short
	Durandal[4]	15.2	4.1	19.3	-
Ideal Rank Syndrome Decoding	BG[14]	0.1	12.6	12.7	Fast
		0.1	10.2	10.3	Short
Ideal Rank Support Learning	BG[14]	0.5	8.4	8.9	Fast
		0.5	6.1	6.6	Short
MinRank	Fen[22]	18.2	9.3	27.5	Fast
		18.2	7.1	25.3	Short
MinRank with Linearized Poly	Fen[22]	18.2	7.2	25.4	Fast
		18.2	5.5	23.7	Short
Rank Syndrome Dec. with Linearized Poly	Fen[22]	0.9	7.4	8.3	Fast
		0.9	5.9	6.8	Short
PKP	Beu[13]	0.1	18.4	18.5	Fast
		0.1	12.1	12.2	Short
	Fen[22]	0.1	16.4	16.5	Fast
		0.1	12.8	12.9	Short
	BG[14]	0.1	9.8	9.9	Fast
		0.1	8.8	8.9	Short
Hash collisions	SPHINCS$^+$[11]	<0.1	16.7	16.7	Fast
		<0.1	7.7	7.7	Short
R-SDP	R-GPS[This work]	0.1	14.8	14.9	Fast
		0.1	12.5	12.6	Short
	R-BG[This work]	0.1	10.8	10.9	Fast
		0.1	9.5	9.6	Short
R-SDP(G)	R-GPS[This work]	0.1	12.7	12.8	Fast
		0.1	11.5	11.6	Short
	R-BG[This work]	0.1	7.7	7.8	Fast
		0.1	7.2	7.3	Short

7 Conclusion

We studied the Restricted Syndrome Decoding Problem (R-SDP), which is similar to the classical SDP with the additional condition that the solution vector has values in a restricted set $\mathbb{E}$. When focusing on full-weight instances, we expect to have a unique solution when $\mathbb{E}$ is sufficiently small; this makes generic solvers more costly than those for SDP, so that we can achieve the same security level with smaller codes. Also, the linear transitive transformations on $\mathbb{E}^n$ are given by a component-wise multiplication with another element of $\mathbb{E}^n$. These two properties are very promising for signature schemes. We have also introduced a new version of the problem, called R-SDP(G), in which only a subgroup of $\mathbb{E}^n$ is employed; this amplifies the positive features of R-SDP and leads to much more compact objects. We adapted some existing zero-knowledge protocols to the use of these new problems. As a result, we are able to achieve signatures in the order of 7 kB, which are highly competitive and compare well with state-of-the-art alternatives.

Appendix

A Representation Technique for R-SDP

In this Appendix, we present a generic solver for the R-SDP, which is an adaption of the BJMM algorithm [9] in combination with the technique of [8] for subsetsum solvers. The security levels provided in this paper are computed taking also this algorithm into account.

For this section, we require some additional notation. For $n \geq \sum_{i=1}^{m} k_i$ we denote by

$$\binom{n}{k_1, \ldots, k_m} = \prod_{i=1}^{m} \binom{\sum_{j=1}^{i} k_j}{k_i} \binom{n}{n - \sum_{i=1}^{m} k_i}$$

the multinomial coefficient. Recall that

$$\lim_{n\to\infty} \log_2 \left(\binom{f(n)}{f_1(n), \ldots, f_m(n)} \right) = F \cdot g_m\left(\tfrac{F_1}{F}, \ldots, \tfrac{F_m}{F}\right),$$

$$\text{with } g_m(x_1, \ldots, x_m) = -\sum_{i=1}^{m} x_i \log_2(x_i) - \left(1 - \sum_{i=1}^{m} x_i\right) \log_2 \left(1 - \sum_{i=1}^{m} x_i\right)$$

and $F = \lim\limits_{n\to\infty} \frac{f(n)}{n}, F_i = \lim\limits_{n\to\infty} \frac{f_i(n)}{n}$ for all $i \in \{1, \ldots, m\}$. Notice that $g_1 = h_2$ corresponds to the binary entropy function.

After the PGE step, explained in Section 3, we are left with solving the smaller instance, i.e., $\mathbf{e}_1\mathbf{H}_2^\top = \mathbf{s}_2$ and $\mathbf{e}_1 \in \mathbb{E}_0^{k+\ell}$ has weight v. The main idea of the BJMM algorithm is to use a sum partition $\mathbf{e}_1 = \mathbf{e}_1^{(1)} + \mathbf{e}_2^{(1)}$. The number of ways in which we can write a vector $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2$, where both the $\mathbf{x}_i$ have to satisfy certain conditions, is called the *number of representations*.

We start with the *representation merge*: given two lists $\mathscr{L}_1, \mathscr{L}_2$ containing $\mathbf{x}_i$ of a certain weight, we add $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2$ to the resulting list $\mathscr{L}$, whenever $\mathbf{x}$ attains some target weight and some syndrome equations are satisfied. These are $\mathbf{x}\mathbf{H}_2^\top = \mathbf{t}$, for either $\mathbf{t} = \mathbf{s}_2$, the target syndrome or $\mathbf{t} = \mathbf{0}$, the zero vector. Assume that for any $\mathbf{x} \in \mathscr{L}$ there are r representations $\{\mathbf{x}_1, \mathbf{x}_2\}$, which lead to the same $\mathbf{x}$. By checking the syndrome equations only on $u = \log_q(r)$ positions, we have with high probability that one representation for each possible $\mathbf{x}$ survives the merge. A representation merge of two lists $\mathscr{L}_1, \mathscr{L}_2$ on u positions costs

$$|\mathscr{L}_1| + |\mathscr{L}_2| + |\mathscr{L}_1| \cdot |\mathscr{L}_2| \, q^{-u}.$$

After the representation merge, one performs a filtering step, which removes vectors which are not well-formed, e.g., do not achieve a given weight constraint or do not live in a desired space. Further steps can then utilize this smaller list.

The representation merge can clearly be used several times, thus we denote by BJMM(a) an algorithm that has *a levels*, where in the first level we do a concatenation merge à la Stern/Dumer. For more details on how exactly the algorithm proceeds, we refer the reader to [29].

Since we have many non-zero entries in our solution, we want many representations of elements in $\mathbb{E}$. For this we have to choose the *search space*, i.e., where $\mathbf{e}_1^{(1)}, \mathbf{e}_2^{(1)}$ live, in a smart way: we want to choose it large enough to gain representations, but small enough to have reasonable list sizes.

To get some fixed entry $x \in \mathbb{E}$ as $x = y + y'$, we could choose $y \in \mathbb{E}, y' \in \mathbb{D} := \{a - b \mid a, b \in \mathbb{E}\} \setminus \{\pm\mathbb{E}_0\}$. If $\mathbb{E}$ has already a lot of additive structure, e.g. when z is even, that is there are many elements $y, y' \in \mathbb{E}$ such that $y + y' \in \mathbb{E}$, then $\mathbb{D}$ becomes small. Thus, we only need a few additional elements in the search space to gain many representations for elements in $\mathbb{E}$. We propose the following search space $X = \mathbb{E} \cup \mathbb{D} \cup -\mathbb{E}$. On each level i, we are considering vectors $\mathbf{x}$ living in X_0, with $v_e^{(i)}$ entries in $\mathbb{E}$, $v_d^{(i)}$ entries in $\mathbb{D}$ and $v_m^{(i)}$ entries in $-\mathbb{E}$.

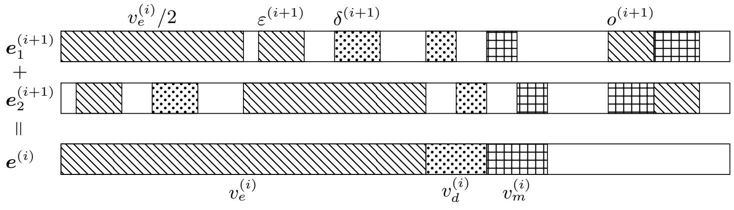

Fig. 5: Counting the number of representations on level i.

To count the number of representations we use Figure 5. We denote by $\varepsilon^{(i+1)}$ the number of entries which are obtained through a $\mathbb{E} + \mathbb{E}$ representation. That is, for a fixed entry x of $\mathbf{e}^{(i)}$, we need to compute the number of possible $y \in \mathbb{E}$ that can reach x, through addition with $\mathbb{E}$:

$$n_e(q, z, x) := |\{y \in \mathbb{E} \mid \exists y' \in \mathbb{E} : y + y' = x \in \mathbb{E}\}| \,.$$

We denote by $\delta^{(i+1)}$ the number of entries of $\mathbf{e}^{(i)}$ obtained through representations $\mathbb{E}+\mathbb{D}$. Hence, for a fixed entry x of $\mathbf{e}^{(i)}$, we need to compute the number of possible $y \in \mathbb{E}$ that can reach x through addition with $\mathbb{D}$:

$$n_d(q,z,x) \coloneqq \left|\{y \in \mathbb{E} \mid \exists y' \in \mathbb{D} : y + y' = x \in \mathbb{E}\}\right| .$$

Since $n_e(q,z,x), n_d(q,z,x)$ are independent of x, we just write $n_e(q,z), n_d(q,z)$. Finally, outside of the support of $\mathbf{e}^{(i)}$, we allow for $o^{(i+1)}$ representations of 0 as $0 = y + (-y) = (-y) + y$, for $y \in \mathbb{E}$. We could also allow for cancellations via $\mathbb{D}$, but as these entries are already only few, they will be optimized to zero.

The vectors $\mathbf{e}_i^{(i+1)}$ have $v_e^{(i+1)} = v_e^{(i)}/2 + \varepsilon^{(i+1)} + o^{(i+1)}$ entries in $\mathbb{E}$, $v_d^{(i+1)} = v_d^{(i)}/2 + \delta^{(i+1)}$ in $\mathbb{D}$ and $v_m^{(i+1)} = v_m^{(i)}/2 + o^{(i+1)}$ in $-\mathbb{E}$. Hence, we get the number of representations

$$\begin{aligned} r^{(i)} =& \binom{v_e^{(i-1)}}{v_e^{(i-1)}/2} \left(\binom{v_e^{(i-1)}/2}{\delta^{(i)}, \varepsilon^{(i)}} n_d(q,z)^{\delta^{(i)}} n_e(q,z)^{\varepsilon^{(i)}} \right)^2 \\ &\cdot \binom{v_d^{(i-1)}}{v_d^{(i-1)}/2} \binom{v_m^{(i-1)}}{v_m^{(i-1)}/2} \binom{k + \ell - v_e^{(i-1)} - v_d^{(i-1)} - v_m^{(i-1)}}{o^{(i)}, o^{(i)}} z^{2o^{(i)}} . \end{aligned} \tag{9}$$

After each merge, the obtained lists are filtered, to get rid of vectors that are not well-formed. After the filtering we are considering vectors in $S^{(i)}$ that have $v_e^{(i)}$ entries in $\mathbb{E}$, $v_d^{(i)}$ entries in $\mathbb{D}$ and $v_m^{(i)}$ entries in $-\mathbb{E}$. Hence,

$$\left|S^{(i)}\right| = \binom{k+\ell}{v_e^{(i)}, v_m^{(i)}, v_d^{(i)}} z^{v_e^{(i)} + v_m^{(i)}} |\mathbb{D}|^{v_d^{(i)}} .$$

To give the asymptotic cost, we need the following notation:

$$\begin{array}{lll} Q = \log_2(q) & V_e^{(i)} = \lim\limits_{n\to\infty} \frac{v_e^{(i)}(n)}{n} & N_e = \lim\limits_{n\to\infty} \frac{1}{n} \log_2(n_e(q,z)) \\ Z = \log_2(z) & V_m^{(i)} = \lim\limits_{n\to\infty} \frac{v_m^{(i)}(n)}{n} & N_d = \lim\limits_{n\to\infty} \frac{1}{n} \log_2(n_d(q,z)) \\ L = \lim\limits_{n\to\infty} \frac{\ell(n)}{n} & V_d^{(i)} = \lim\limits_{n\to\infty} \frac{v_d^{(i)}(n)}{n} & \varSigma^{(i)} = \lim\limits_{n\to\infty} \frac{1}{n} \log_2\left(\left|S^{(i)}\right|\right) \\ U^{(i)} = Q \lim\limits_{n\to\infty} \frac{u^{(i)}(n)}{n} & \varDelta = \lim\limits_{n\to\infty} \frac{1}{n} \log_2(|\mathbb{D}|) & \\ D^{(i)} = \lim\limits_{n\to\infty} \frac{\delta^{(i)}(n)}{n} & E^{(i)} = \lim\limits_{n\to\infty} \frac{\varepsilon^{(i)}(n)}{n} & O^{(i)} = \lim\limits_{n\to\infty} \frac{o^{(i)}(n)}{n} \end{array}$$

Theorem 3. *The presented BJMM(3) algorithm has a cost of* $2^{nF(R,q,z,\omega)}$, *where*

$$F(R,q,z,W) = N(R,q,z,W) + C(R,q,z,W),$$

where $N(R,q,z,W)$ *denotes the number of iterations and is given by*

$$h_2(W) - (R+L)h_2\left(\tfrac{V}{R+L}\right) - (1-R-L)h_2\left(\tfrac{W-V}{1-R-L}\right)$$

and $C(R,q,z,W)$ denotes the cost of one iteration, which is given by

$$\max\left\{\Sigma^{(2)}/2, \Sigma^{(2)} - U^{(2)}, 2\Sigma^{(2)} - U^{(2)} - U^{(1)}, 2\Sigma^{(1)} - U^{(1)} - LQ\right\},$$

where for $i \in \{1,2\}$ and $V_e^{(0)} = V$, $V_d^{(0)} = V_m^{(0)} = 0$ we set

$$\begin{aligned}
U^{(i)} &= R + L - R^{(i-1)} + R^{(i-1)} h_2\left(\tfrac{2O^{(i)}}{R^{(i-1)}}\right) + O^{(i)} \\
&\quad + V_e^{(i-1)} g_2\left(\tfrac{2E^{(i)}}{V_e^{(i-1)}}, \tfrac{2D^{(i)}}{V_e^{(i-1)}}\right) + 2\left(D^{(i)} N_d + E^{(i)} N_e + O^{(i)} Z\right), \\
\Sigma^{(i)} &= (R+L) g_3\left(\tfrac{V_e^{(i)}}{R+L}, \tfrac{V_m^{(i)}}{R+L}, \tfrac{V_d^{(i)}}{R+L}\right) + \left(V_e^{(i)} + V_m^{(i)}\right) Z + V_d^{(i)} \Delta, \\
R^{(i)} &= R + L - V_e^{(i)} - V_d^{(i)} - V_m^{(i)}, \\
V_e^{(i)} &= V_e^{(i-1)}/2 + E^{(i)} + O^{(i)},\ V_d^{(i)} = V_d^{(i-1)}/2 + D^{(i)},\ V_m^{(i)} = V_m^{(i-1)}/2 + O^{(i)}.
\end{aligned}$$

A.1 Refinements

For large weight vectors, it makes sense to first shift the considered instance. That is for a fixed $c \in \mathbb{F}_q$, we shift the whole error set $\mathbb{E}$ to $\widetilde{\mathbb{E}} = \{a + c \mid a \in \mathbb{E}\}$. Let us denote by $\mathbf{c}$ the all c vector. Then, such shifting can easily be done by computing the syndrome $\mathbf{s}_c$ of $\mathbf{c}$ and adding it to the original syndrome $\mathbf{s}$: $(\mathbf{e} + \mathbf{c})\mathbf{H}^\top = \mathbf{s} + \mathbf{s}_c$. By choosing $c \in -\mathbb{E}$, one can set the error at all positions with value c to zero. Hence, one obtains $\widetilde{\mathbb{E}} = \{a + c \mid a \in \mathbb{E}\} \setminus \{0\}$ of size $\widetilde{z} = z - 1$. With this error set of reduced size, one can proceed as before. That is we again use the sets

$$\widetilde{\mathbb{D}} = \left\{a - b \mid a, b \in \widetilde{\mathbb{E}}\right\} \setminus \left\{\pm\widetilde{\mathbb{E}}_0\right\} \text{ and } -\widetilde{\mathbb{E}} = \left\{-e \mid e \in \widetilde{\mathbb{E}}\right\} \setminus \widetilde{\mathbb{E}}.$$

Note that for these sets, $n_e(q,z,x)$ and $n_d(q,z,x)$ are indeed dependent on the element x. In order to avoid a more complicated analysis, we resolve this issue by defining the *average* number of representations for an element in $\widetilde{\mathbb{E}}$ as

$$\widetilde{n_e}(q,z,c) = \frac{1}{\widetilde{z}} \sum_{x \in \widetilde{\mathbb{E}}} n_e(q,z,x) \quad \text{and} \quad \widetilde{n_d}(q,z,c) = \frac{1}{\widetilde{z}} \sum_{x \in \widetilde{\mathbb{E}}} n_d(q,z,x),$$

which depends not on the particular element but only on the chosen shift. Hence, $\widetilde{n_e}$ and $\widetilde{n_d}$ can be directly used in Theorem 3.

In Figure 6, we compare the complexity coefficients of different information set decoders as a function of the relative error weight W. The considered code rate is $R = 0.45$. The field size $q = 157$ allows for $z = 12$ and $z = 13$, which correspond to the solid and dashed lines, respectively. While the performance of Stern depends only on the size of $\mathbb{E}$, the performance of the BJMM algorithms depends on its structure. For $z = 12$, $\mathbb{E}$ possesses a lot of additive structure, which is why BJMM(3) can improve over Stern. In particular, $\mathbb{E} = -\mathbb{E}$ and $\alpha(157, 12) = 2$ allow for an increased number of representations. This is not

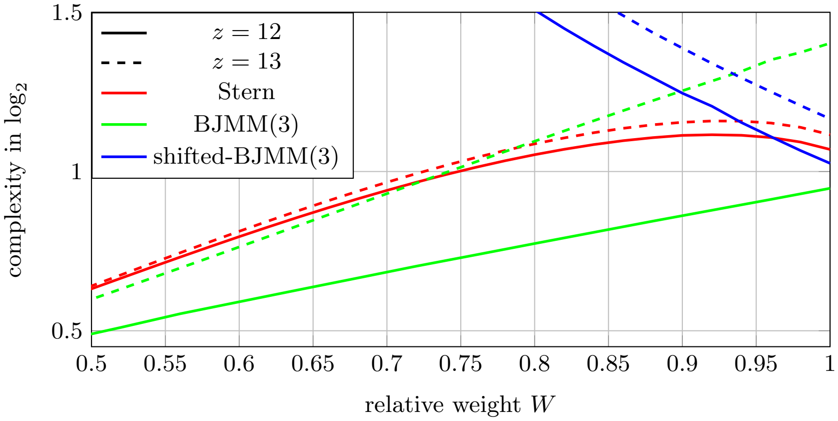

Fig. 6: Comparison of the asymptotic complexity for the restricted Stern/Dumer algorithm and the restricted BJMM algorithm.

the case for $z = 13$, where we only improve over Stern in the low error weight regime. Finally, we observe that shifting has to be take into account for high error weights, but becomes quickly impractical as the weight decreases. Taking these observations into account, we avoid choosing instances for which the BJMM algorithm can achieve a significantly lower complexity than restricted Stern.

B NP-completeness for the Submatrix Rank Problem

We reduce from the well known NP-complete problem of establishing whether a given code $\mathscr{C} = \langle \mathbf{G} \rangle$, for $\mathbf{G} \in \mathbb{F}_z^{k \times n}$, has codewords with weight $\leq d$. Due to the Singleton bound, we focus on $d < n - k + 1$. We show that any pair $\{\mathbf{G}, d\}$ can be transformed, in polynomial time, into an instance of the Submatrix Rank Problem. We will denote by Solve an algorithm that, on input a matrix $\mathbf{M}_G \in \mathbb{F}_z^{m \times n}$ and two integers $t, m' \in \mathbb{N}$, returns "YES" if $J \subseteq \{1, \ldots, n\}$ of size t and $m' = \mathrm{rk}((\mathbf{M}_G)_J)$ exists, and "NO" otherwise. We can set $\mathbf{M}_G := \mathbf{G}$ and $t := n - d$. Notice that $m := k$ and $t > n - (n - k + 1) = m - 1$. We run Solve for all $m' \leq m - 1$.

- Assume that the call for $m^\star$ on Solve returns a "YES". Since $t \geq m$, we apply thesis ii) of Theorem 1 and learn that $\mathscr{C} = \langle \mathbf{G} \rangle$ has a subcode $\mathscr{C}'$ of dimension $m - m^\star$ with support size $s \leq n - t = d$. Since $d(\mathscr{C}) \leq d(\mathscr{C}') \leq s \leq d$, we return "YES" for the original problem.
- Assume that none of the calls on Solve return a "YES", then all subcodes have a support size greater than $t - n = d$. Notice that we also tried $m' = m - 1$, so Solve has also considered existence of subcodes of dimension $m - m' = 1$, that is, codewords. So, we return "NO" for the original problem.

References

[1] C. Aguilar, P. Gaborit, and J. Schrek. “A new zero-knowledge code based identification scheme with reduced communication”. In: *2011 IEEE Information Theory Workshop*. IEEE. 2011, pp. 648–652.

[2] C. Aguilar-Melchor et al. “The Return of the SDitH”. In: *Cryptology ePrint Archive* (2022).

[3] N. Aragon et al. “Cryptanalysis of a code-based full-time signature”. In: *Designs, Codes and Cryptography* 89 (2021), pp. 2097–2112.

[4] N. Aragon et al. “Durandal: a rank metric based signature scheme”. In: *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III 38*. Springer. 2019, pp. 728–758.

[5] M. Baldi et al. “A new path to code-based signatures via identification schemes with restricted errors”. In: *arXiv preprint arXiv:2008.06403* (2020).

[6] A. Barenghi et al. “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*. Springer. 2021, pp. 23–43.

[7] S. Barg. “Some new NP-complete coding problems”. In: *Problemy Peredachi Informatsii* 30.3 (1994), pp. 23–28.

[8] A. Becker, J.-S. Coron, and A. Joux. “Improved generic algorithms for hard knapsacks”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2011, pp. 364–385.

[9] A. Becker et al. “Decoding random binary linear codes in $2^{n/20}$: How 1+ 1= 0 improves information set decoding”. In: *Advances in Cryptology–EUROCRYPT 2012: 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings 31*. Springer. 2012, pp. 520–536.

[10] E. Berlekamp, R. McEliece, and H. Van Tilborg. “On the inherent intractability of certain coding problems (corresp.)” In: *IEEE Transactions on Information Theory* 24.3 (1978), pp. 384–386.

[11] D. J. Bernstein et al. “The SPHINCS+ signature framework”. In: *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 2019, pp. 2129–2146.

[12] S. Bettaieb et al. “Zero-Knowledge Reparation of the Véron and AGS Code-based Identification Schemes”. In: *2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE. 2021, pp. 55–60.

[13] W. Beullens. “Sigma protocols for MQ, PKP and SIS, and fishy signature schemes”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2020, pp. 183–211.

[14] L. Bidoux and P. Gaborit. “Shorter Signatures from Proofs of Knowledge for the SD, MQ, PKP and RSD Problems”. In: *arXiv preprint arXiv:2204.02915* (2022).

[15] L. Bidoux, P. Gaborit, and N. Sendrier. “Quasi-Cyclic Stern Proof of Knowledge”. In: *arXiv preprint arXiv:2110.05005* (2021).

[16] L. Bidoux et al. “Code-based signatures from new proofs of knowledge for the syndrome decoding problem”. In: *Designs, Codes and Cryptography* (2022), pp. 1–48.

[17] P.-L. Cayrel, P. Véron, and S. M. E. Yousfi Alaoui. “A zero-knowledge identification scheme based on the q-ary syndrome decoding problem”. In: *International Workshop on Selected Areas in Cryptography*. Springer. 2010, pp. 171–186.

[18] T. Debris-Alazard, N. Sendrier, and J.-P. Tillich. “Wave: A new code-based signature scheme”. In: (2018).

[19] J.-C. Deneuville and P. Gaborit. “Cryptanalysis of a code-based one-time signature”. In: *Designs, Codes and Cryptography* 88 (2020), pp. 1857–1866.

[20] I. I. Dumer. “Two decoding algorithms for linear codes”. In: *Problemy Peredachi Informatsii* 25.1 (1989), pp. 24–32.

[21] J.-C. Faugere et al. “A distinguisher for high-rate McEliece cryptosystems”. In: *IEEE Transactions on Information Theory* 59.10 (2013), pp. 6830–6844.

[22] T. Feneuil. “Building MPCitH-based Signatures from MQ, MinRank, Rank SD and PKP”. In: *Cryptology ePrint Archive* (2022).

[23] T. Feneuil, A. Joux, and M. Rivain. “Shared permutation for syndrome decoding: New zero-knowledge protocol and code-based signature”. In: *Designs, Codes and Cryptography* (2022), pp. 1–46.

[24] T. Feneuil, A. Joux, and M. Rivain. “Syndrome decoding in the head: Shorter signatures from zero-knowledge proofs”. In: *Cryptology ePrint Archive* (2022).

[25] A. Fiat and A. Shamir. “How to Prove Yourself: Practical Solutions to Identification and Signature Problems.” In: *Crypto.* Vol. 86. Springer. 1986, pp. 186–194.

[26] S. Gueron, E. Persichetti, and P. Santini. “Designing a Practical Code-Based Signature Scheme from Zero-Knowledge Proofs with Trusted Setup”. In: *Cryptography* 6.1 (2022), p. 5.

[27] Y. Ishai et al. “Zero-knowledge from secure multiparty computation”. In: *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing.* 2007, pp. 21–30.

[28] D. Kales and G. Zaverucha. “An attack on some signature schemes constructed from five-pass identification schemes”. In: *Cryptology and Network Security: 19th International Conference, CANS 2020, Vienna, Austria, December 14–16, 2020, Proceedings.* Springer. 2020, pp. 3–22.

[29] A. Meurer. “A coding-theoretic approach to cryptanalysis”. PhD thesis. Ruhr-Universität Bochum, 2013.

[30] A. Otmani and J.-P. Tillich. “An efficient attack on all concrete KKS proposals”. In: *Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29–December 2, 2011. Proceedings 4.* Springer. 2011, pp. 98–116.

[31] P. Santini, M. Baldi, and F. Chiaraluce. “Computational Hardness of the Permuted Kernel and Subcode Equivalence Problems”. In: *Cryptology ePrint Archive* (2022).

[32] P. Santini, M. Baldi, and F. Chiaraluce. “Cryptanalysis of a one-time code-based digital signature scheme”. In: *2019 IEEE International Symposium on Information Theory (ISIT).* IEEE. 2019, pp. 2594–2598.

[33] A. Shamir. “An efficient identification scheme based on permuted kernels”. In: *Advances in Cryptology — CRYPTO’ 89 Proceedings. CRYPTO 1989.* Ed. by G. Brassard. Vol. 435. Lecture Notes in Computer Science. Springer, 1989, pp. 606–609.

[34] J. Stern. “A method for finding codewords of small weight”. In: *International Colloquium on Coding Theory and Applications.* Springer. 1988, pp. 106–113.

[35] J. Stern. “A new identification scheme based on syndrome decoding”. In: *Annual International Cryptology Conference*. Springer. 1993, pp. 13–21.

[36] J. Stern. “Designing identification schemes with keys of short size”. In: *Advances in Cryptology—CRYPTO'94: 14th Annual International Cryptology Conference Santa Barbara, California, USA August 21–25, 1994 Proceedings*. Springer. 2001, pp. 164–173.

[37] P. Véron. “Improved identification schemes based on error-correcting codes”. In: *Applicable Algebra in Engineering, Communication and Computing* 8.1 (1997), pp. 57–69.

[38] V. Weger et al. “On the hardness of the Lee syndrome decoding problem”. In: *Advances in Mathematics of Communications* (2022).