Blockwise Rank Decoding Problem and LRPC Codes: Cryptosystems with Smaller Sizes

Yongcheng Song[1], Jiang Zhang[1(✉)], Xinyi Huang[2], and Wei Wu[3,4]

[1] State Key Laboratory of Cryptology, P. O. Box 5159, Beijing 100878, China
jiangzhang09@gmail.com

[2] Artificial Intelligence Thrust, Information Hub, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511455, China
xinyi@ust.hk

[3] College of Education Sciences, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511455, China
serenaweiwu@hkust-gz.edu.cn

[4] School of Mathematics and Statistics, Fujian Normal University, Fuzhou 350117, China

Abstract. In this paper, we initiate the study of the Rank Decoding (RD) problem and LRPC codes with blockwise structures in rank-based cryptosystems. First, we introduce the blockwise errors (ℓ-errors) where each error consists of ℓ blocks of coordinates with disjoint supports, and define the blockwise RD (ℓ-RD) problem as a natural generalization of the RD problem whose solutions are ℓ-errors (note that the standard RD problem is actually a special ℓ-RD problem with $\ell = 1$). We adapt the typical attacks on the RD problem to the ℓ-RD problem, and find that the blockwise structures do not ease the problem too much: the ℓ-RD problem is still exponentially hard for appropriate choices of $\ell > 1$. Second, we introduce blockwise LRPC (ℓ-LRPC) codes as generalizations of the standard LPRC codes whose parity-check matrices can be divided into ℓ sub-matrices with disjoint supports, i.e., the intersection of two subspaces generated by the entries of any two sub-matrices is a null space, and investigate the decoding algorithms for ℓ-errors. We find that the gain of using ℓ-errors in decoding capacity outweighs the complexity loss in solving the ℓ-RD problem, which makes it possible to design more efficient rank-based cryptosystems with flexible choices of parameters.

As an application, we show that the two rank-based cryptosystems submitted to the NIST PQC competition, namely, RQC and ROLLO, can be greatly improved by using the ideal variants of the ℓ-RD problem and ℓ-LRPC codes. Concretely, for 128-bit security, our RQC has total public key and ciphertext sizes of 2.5 KB, which is not only about 50% more compact than the original RQC, but also smaller than the NIST Round 4 code-based submissions HQC, BIKE, and Classic McEliece.

Keywords: Post-Quantum Cryptography · NIST PQC Candidates · Rank Metric Code-Based Cryptography · Rank Decoding Problem · LRPC Codes

J. Guo and R. Steinfeld (Eds.): ASIACRYPT 2023, LNCS 14444, pp. 284–316, 2023.
https://doi.org/10.1007/978-981-99-8739-9_10

1 Introduction

Since traditional cryptographic schemes based on number theoretic assumptions are at risk from the possible attacks using quantum computers, the design of post-quantum cryptosystems, such as code-based cryptosystems, has become the consensus of industry and academia. Last year, three code-based cryptosystems using the Hamming metric codes, namely, BIKE, Classic McEliece, and HQC, had been selected to the fourth round of NIST post-quantum standardization process for future standardization [35]. As a nice alternative to Hamming metric code-based cryptography, code-based cryptography using the rank metric, namely, rank-based cryptography, is more efficient in computational efficiency and bandwidth, and deserves further research as encouraged by NIST [34].

$\mathbb{F}_{q^m}$-Linear Codes with Rank Metric and Rank Decoding Problem. Codes used in rank-based cryptography are $\mathbb{F}_{q^m}$-linear codes with rank metric over a degree m extension field $\mathbb{F}_{q^m}$ of $\mathbb{F}_q$. Let $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m) \in \mathbb{F}_{q^m}^m$ be a basis of $\mathbb{F}_{q^m}$ viewed as an m-dimensional vector space over $\mathbb{F}_q$. Then, any $\boldsymbol{e} = (e_1, e_1, \ldots, e_n) \in \mathbb{F}_{q^m}^n$ has an associated matrix $\mathrm{Mat}(\boldsymbol{e}) \in \mathbb{F}_q^{m \times n}$ such that $\boldsymbol{e} = \boldsymbol{\alpha}\mathrm{Mat}(\boldsymbol{e})$. *The rank weight* $\|\boldsymbol{e}\|_{\mathrm{R}}$ of $\boldsymbol{e}$ is defined as the rank of $\mathrm{Mat}(\boldsymbol{e})$. *The support* $\mathrm{Supp}(\boldsymbol{e})$ of $\boldsymbol{e}$ is the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}$ spanned by the coordinates of $\boldsymbol{e}$. It follows from definition that $\|\boldsymbol{e}\|_{\mathrm{R}}$ equals to the dimension of $\mathrm{Supp}(\boldsymbol{e})$. The set of errors of length n and weight r is denoted by $\mathcal{S}_r^n$. An $\mathbb{F}_{q^m}$-linear code ($[n,k]_{q^m}$) with rank metric of length n and dimension k is a dimension k subspace of $\mathbb{F}_{q^m}^n$, which can be represented by a generator matrix of size $k \times n$ or a parity-check matrix of size $(n-k) \times n$ over $\mathbb{F}_{q^m}$.

Let $\boldsymbol{G}$ be the generator matrix of a random $[n,k]_{q^m}$-linear code, $\boldsymbol{y} \in \mathbb{F}_{q^m}^n$, and $r \in \mathbb{N}$. The Rank Decoding (RD) problem is to find $\boldsymbol{x} \in \mathbb{F}_{q^m}^k$ and $\boldsymbol{e} \in \mathcal{S}_r^n$ such that $\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e}$. Although the RD problem is not shown to be NP-hard, it is very close to the Hamming metric decoding problem which is NP-hard [23], and can be seen as a structured version of the MinRank problem which is also NP-hard [17]. Moreover, after more than three decades of study, the best known algorithms for solving the RD problem are all exponential. This makes the RD problem a promising hard problem to construct secure cryptosystems.

Rank-Based Cryptography. The first rank-based cryptosystem, known as the GPT cryptosystem [19], was based on Gabidulin codes [18] which have analogous structures to Reed-Solomon codes. The GPT cryptosystem and its early variants were broken by Overbeck attack [38], in the much same way as McEliece schemes based on Reed-Solomon codes were attacked in [16,39]. The recent variant [28] was analyzed with some insecure parameters region being found in [15,24]. As these attacks [15,16,24,38,39] mainly expose the security flaws of the GPT cryptosystem by exploiting the strong algebraic structure of Gabidulin codes, it is still possible to construct secure and efficient rank-based cryptosystems.

A very significant step was using the Low Rank Parity Check (LRPC) codes [4,20] and the Gabidulin codes to build cryptosystems [2,20,22,29,30], which can be viewed as rank metric analogues of the MDPC cryptosystem [33], NTRU [25], or Alekhnovich [1]. Four cryptosystems of this kind, namely, RQC [30],

Lake, Locker [29], and Ouroboros-R [2], were submitted to the NIST PQC standardization process in 2017, with the latter three being merged into ROLLO in the second round. The combinatorial attacks [5,21,37] were once considered to be the most efficient attacks against the parameters region of RQC and ROLLO. However, it turned out later that the improved dedicated algebraic attacks [7,9] could greatly reduce the concrete security of RQC and ROLLO. This is the main reason that RQC and ROLLO were not selected to the third round of the NIST PQC standardization process. New parameter sets [2,29,30] for RQC and ROLLO were proposed to provide adequate security against algebraic attacks. As the new key and ciphertext sizes of RQC and ROLLO remain competitive, NIST encourages further research on rank-based cryptography [34].

1.1 Our Contribution

We initiate the study of the RD problem and LRPC codes with blockwise structures to design secure and efficient rank-based cryptosystems. First, we introduce the blockwise errors (ℓ-errors) where each error consists of ℓ blocks of coordinates with disjoint supports, and define the blockwise RD (ℓ-RD) problem as a natural generalization of the RD problem whose solutions are ℓ-errors. Notably, the standard RD problem can be seen as a special ℓ-RD problem with $\ell = 1$, or equivalently the ℓ-RD problem can be treated as a structured RD problem. Since the attacks may benefit from the blockwise structure, the ℓ-RD problem is inherently not harder than the standard one. Fortunately, this structure does not ease the problem too much: we only observe a reduction about ℓ times in the exponent to solve the ℓ-RD problem by carefully examining the typical attacks for the standard RD problem, implying that the ℓ-RD problem is still exponentially hard for appropriate choices of constant $\ell > 1$.

Second, we introduce the blockwise LRPC (ℓ-LRPC) codes as generalizations of the standard LPRC codes whose parity-check matrices can be divided into ℓ sub-matrices with disjoint supports, i.e., the intersection of two subspaces generated by the entries of any two sub-matrices is a null space, and investigate the decoding algorithms for ℓ-errors. We find that the decoding algorithm can also benefit from the blockwise structure: the decoding capacity can be significantly improved by a factor of ℓ. In particular, a suitably defined $[n, k]_{q^m}$ ℓ-LRPC code can actually decode an ℓ-error with weight up to $(n - k)/2$, which achieves the decoding capacity of rank codes of optimal distance. This makes it possible to design more efficient rank-based cryptosystems with flexible choices of parameters, by making a tradeoff between the hardness of the ℓ-RD problem and the decoding capacity of the ℓ-LPRC codes.

Finally, we show that the two rank-based cryptosystems submitted to the NIST PQC competition, namely, RQC and ROLLO, can be greatly improved by using the ideal variants of the ℓ-RD problem and ℓ-LRPC codes. Concretely, for 128-bit security, our RQC has total public key and ciphertext sizes of 2.5 KB, which is not only about 50% more compact than the original RQC, but also smaller than the NIST Round 4 code-based submissions HQC, BIKE, and Classic McEliece. A detailed comparison with related works is given in Subsect. 1.2.

1.2 Technical Overview

Recall that the set of errors of length n and weight r is denoted by $\mathcal{S}_r^n$. By definition, all n coordinates of an error $\boldsymbol{e} \in \mathcal{S}_r^n$ belong to the same support of dimension r. In particular, let $\boldsymbol{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_r) \in \mathbb{F}_{q^m}^r$ be a basis of the support $\mathrm{Supp}(\boldsymbol{e})$, then there is an $r \times n$ coefficient matrix $\boldsymbol{C}$ such that $\boldsymbol{e} = \boldsymbol{\varepsilon}\boldsymbol{C}$.

The Blockwise Errors (ℓ-errors). Let $\boldsymbol{n} = (n_1, \ldots, n_\ell)$ and $\boldsymbol{r} = (r_1, \ldots, r_\ell)$ be vectors of positive integers. We say that an error $\boldsymbol{e} \in \mathcal{S}_r^n$ with $n = \sum_{i=1}^{\ell} n_i$ and $r = \sum_{i=1}^{\ell} r_i$ is an ℓ-error with parameters $\boldsymbol{n}$ and $\boldsymbol{r}$ if it can be divided into ℓ sub-vectors $\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_\ell)$ such that 1) the sub-vector $\boldsymbol{e}_i \in \mathbb{F}_{q^m}^{n_i}$ has weight r_i for all $i \in \{1..\ell\}$; and 2) the supports of these sub-vectors are mutually disjoint, namely, $\mathrm{Supp}(\boldsymbol{e}_i) \cap \mathrm{Supp}(\boldsymbol{e}_j) = \{0\}$ for all $i \neq j$. Denote $\mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ as the set of blockwise errors with parameters $\boldsymbol{n}$ and $\boldsymbol{r}$. By definition, the set $\mathcal{S}_r^n$ is exactly the set $\mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ of ℓ-errors with $\ell = 1$. For $\ell > 1$, $\mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ is a proper subset of $\mathcal{S}_r^n$. In particular, for any $\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_\ell) \in \mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$, if we let $\boldsymbol{\varepsilon}_i = (\varepsilon_{i1}, \varepsilon_{i2}, \ldots, \varepsilon_{ir_i}) \in \mathbb{F}_{q^m}^{r_i}$ be a basis of $\mathrm{Supp}(\boldsymbol{e}_i)$, then the coefficient matrix $\boldsymbol{C}$ of $\boldsymbol{e}$ w.r.t. the basis $\boldsymbol{\varepsilon} = (\boldsymbol{\varepsilon}_1, \boldsymbol{\varepsilon}_2, \ldots, \boldsymbol{\varepsilon}_\ell)$, i.e., $\boldsymbol{e} = \boldsymbol{\varepsilon}\boldsymbol{C}$, has a special block-diagonal form:

$$\boldsymbol{C} = \begin{pmatrix} \boldsymbol{C}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_2 & \boldsymbol{0} & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{C}_\ell \end{pmatrix} \in \mathbb{F}_q^{r \times n} \tag{1}$$

where $\boldsymbol{e}_i = \varepsilon_i \boldsymbol{C}_i$. As we will show later, the attacks can benefit from the block-diagonal structure.

The Blockwise RD (ℓ-RD) Problem. We define the ℓ-RD problem as a natural generalization of the RD problem whose solutions are ℓ-errors. Recall that the RD problem asks an algorithm given as inputs a generator matrix $\boldsymbol{G}$ of random $[n,k]_{q^m}$-linear code $\mathcal{C}$, a vector $\boldsymbol{y} \in \mathbb{F}_{q^m}^n$, and an integer $r \in \mathbb{N}$, outputs $\boldsymbol{x} \in \mathbb{F}_{q^m}^k$ and $\boldsymbol{e} \in \mathcal{S}_r^n$ such that $\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e}$. The RD problem can be solved by finding a codeword $\boldsymbol{e} \in \mathcal{S}_r^n$ in the $[n, k+1]_{q^m}$ extended code $\mathcal{C}_{\boldsymbol{y}} = \mathcal{C} + \langle \boldsymbol{y} \rangle$ of $\mathcal{C}$. Let $\boldsymbol{H}_{\boldsymbol{y}} \in \mathbb{F}_{q^m}^{(n-k-1)\times n}$ be the parity-check matrix of $\mathcal{C}_{\boldsymbol{y}}$. The problem can be further reduced to find an $\boldsymbol{e} \in \mathcal{S}_r^n$ such that $\boldsymbol{e}\boldsymbol{H}_{\boldsymbol{y}}^\top = \boldsymbol{\varepsilon}\boldsymbol{C}\boldsymbol{H}_{\boldsymbol{y}}^\top = \boldsymbol{0}$.

There are two main kinds of attacks for the RD problem, i.e., combinatorial attacks [5,14,21,37] and algebraic attacks [7–9,21]. The basic idea of the combinatorial attacks [5,14,21,37] is to guess some unknown variables about the equations $\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e}$ or $\boldsymbol{e}\boldsymbol{H}_{\boldsymbol{y}}^\top = \boldsymbol{\varepsilon}\boldsymbol{C}\boldsymbol{H}_{\boldsymbol{y}}^\top = \boldsymbol{0}$ so that they can be directly solved by using Gaussian eliminations (note that number of equations are much less than that of the variables). The guess complexity is the main cost for the combinatorial attacks. In contrast, the algebraic attacks [7–9,21] resort to establish sufficiently more equations using different algebraic properties such as the annulator polynomial, so that the error $\boldsymbol{e}$ can be directly found by solving those

equations. The complexity of the algebraic attacks is mainly determined by the number of the unknown variables of those equations. By carefully investigating the typical attacks, we find that both combinatorial and algebraic attacks can benefit from the blockwise structures, the basic reason is that the coefficient matrix $\boldsymbol{C}$ for an ℓ-error has a special block-diagonal form, which allows to greatly reduce the number of the unknown variables. The take-away message is that the best cost for solving the ℓ-RD problem is roughly equal to the ℓ-th square root of the cost for solving the standard RD problem (with the same parameters). This means that for appropriate choices of constant $\ell > 1$ such as $\ell = 2$ or 3 in our applications, the ℓ-RD problem is still exponentially hard.

The Blockwise LRPC (ℓ-LRPC) Codes. Let $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ be the parity-check matrix of an $[n,k]_{q^m}$ LRPC code. The entries of $\boldsymbol{H}$ generate an $\mathbb{F}_q$-linear subspace F of dimension d (for simplicity, we call $\boldsymbol{H}$ a matrix of weight d and support F). Let $\boldsymbol{e} \in \mathcal{S}_r^n$ be an error of support E and let $\boldsymbol{s} = \boldsymbol{H}\boldsymbol{e}^\top$. Let EF be the product space of E and F, whose dimension is equal to rd with overwhelming probability when rd is sufficiently smaller than m. The decoding algorithm works by first recovering the product space EF using the support $\mathrm{Supp}(\boldsymbol{s})$ of $\boldsymbol{s}$ (which requires the weight $\|\boldsymbol{s}\|_{\mathrm{R}}$ is equal to the dimension of EF), then recovering the error support E from EF, and finally solving the linear equations $\boldsymbol{s} = \boldsymbol{H}\boldsymbol{e}^\top$ using E. The Decode Failure Rate (DFR) is about $q^{\|\boldsymbol{s}\|_{\mathrm{R}}-(n-k)} = q^{rd-(n-k)}$, implying that an LPRC code of weight d can decode errors of weight up to $\frac{n-k}{d}$.

We define the blockwise LRPC (ℓ-LRPC) codes as generalizations of the standard LPRC codes whose parity-check matrices can be divided by columns into ℓ sub-matrices with disjoint supports. Let $\boldsymbol{n} = (n_1, \ldots, n_\ell)$ and $\boldsymbol{d} = (d_1, \ldots, d_\ell)$ be vectors of positive integers and $k \in \mathbb{N}$. We say that an $[n,k]_{q^m}$ LRPC code is an ℓ-LRPC code with parameters $n = \sum_{i=1}^{\ell} n_i$ and $d = \sum_{i=1}^{\ell} d_i$ if its parity-check matrix $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ can be divided into ℓ sub-matrices $\boldsymbol{H} = (\boldsymbol{H}_1, \boldsymbol{H}_2, \cdots, \boldsymbol{H}_\ell)$ such that 1) the sub-matrix $\boldsymbol{H}_i \in \mathbb{F}_{q^m}^{(n-k)\times n_i}$ has small weight d_i for all $i \in \{1..\ell\}$; and 2) the supports $\{F_i = \mathrm{Supp}(\boldsymbol{H}_i)\}$ of these sub-matrices are mutually disjoint, namely, $F_i \cap F_j = \{0\}$ for all $i \neq j$.

The decoding algorithm for ℓ-LRPC codes works the same way as the one for standard LRPC codes. For traditional errors, an ℓ-LRPC code has the same decoding capacity as a standard LRPC code. However, it is more powerful when decoding ℓ-errors. This is because for an ℓ-error $\boldsymbol{e} \in \mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ with supports $(E_1, E_2, \ldots, E_\ell)$ and $\boldsymbol{r} = (r_1, \ldots, r_\ell)$, the product space in consideration becomes $\sum_{i=1}^{\ell} E_i F_i$, whose dimension is upper bounded by $\sum_{i=1}^{\ell} r_i d_i < rd$, where $r = \sum_{i=1}^{\ell} r_i$. This means that the ℓ-LRPC code can decode an ℓ-error with a much larger weight r. Formally, we have the following Theorem 1.1 (see the proofs in Sect. 4).

Theorem 1.1. *When $d_1 = d_2 = \cdots = d_\ell$, the ℓ-LRPC code allows to decode ℓ-errors of weight up to $r = \sum_{j=1}^{\ell} r_j = \frac{n-k}{d_1}$. By setting $d_1 = d_2 = \cdots = d_\ell = 2$, it can decode ℓ-errors of weight up to $\frac{n-k}{2}$.*

Theorem 1.1 implies that when dealing with ℓ-errors, the decoding capacity for the ℓ-LRPC codes is ℓ times larger than that of the standard LRPC codes. For example, fixing $d = 4$, $r = 8$, and the DFR of q^{32-n-k}, an $[n, k]_{q^m}$ LRPC code can decode errors of weight 8, but an $[n, k]_{q^m}$ 2-LRPC codes with parameter $\boldsymbol{d} = (d_1, d_2) = (2, 2)$ can decode ℓ-errors with parameter $\boldsymbol{r} = (r_1, r_2) = (8, 8)$ of weight up to $r = r_1 + r_2 = 16$.

Applications. By making a tradeoff between the hardness of the ℓ-RD problem and the decoding capacity of the ℓ-LRPC codes, it is possible to design more efficient and secure rank-based cryptosystems with flexible choices of parameters. In particular, the blockwise structures would lead to larger parameters to reserve the security, but the gain in decoding capacity still allows us to design more efficient cryptosystems. As an application, we show that both RQC and ROLLO cryptosystems can be greatly improved by using the ideal variants of the ℓ-RD problem and ℓ-LRPC codes. A brief comparison with related coded-based cryptosystems at the same 128-bit security is summarized in Table 1, which shows that our RQC is about 50% more compact than the original RQC, and has smaller sizes than the three code-based cryptosystems using the Hamming metric, namely, HQC, BIKE, and Classic McEliece.

Table 1. Comparisons of size and DFR for 128-bit security.

Schemes		pks (bytes)	cts (bytes)	total (bytes)	DFR
RQC	**Our**	860	1704	2564	–
	NIST [30]	1834	3652	5486	–
Lake (ROLLO-I)	**Our**	511	511	1022	2^{-31}
	NIST [29]	696	696	1392	2^{-28}
Locker (ROLLO-II)	**Our**	1814	1942	3756	2^{-131}
	NIST [29]	1941	2089	4030	2^{-134}
Ouroboros-R (ROLLO-III)	**Our**	623	1166	1789	2^{-33}
	TIT 2022 [2]	736	1431	2167	2^{-28}
HQC	NIST [31]	2249	4497	6746	–
BIKE	NIST [1]	1541	1573	3114	2^{-128}
Classic McEliece	NIST [10]	261120	96	261216	–
Ouroboros	TIT 2022 [2]	1566	3100	4666	2^{-128}

The public key size (pks), the ciphertext size (cts), total = pks+cts.

1.3 Other Related Works

The idea of using blockwise errors can be seen as an adaption of the LPN/LWE problem in rank metric [11]. Our blockwise codes are also related to the sum-rank metric codes [13], where the error is also divided into ℓ blocks and the sum-rank weight is defined as the sum of rank weight of each block. One main difference is that we explicitly require the ℓ blocks to have disjoint supports, which is very crucial for our results in this paper.

1.4 Organization

After some notations given in Sect. 2, we define the ℓ-errors and analyze the complexity of solving the ℓ-RD problem in Sect. 3. Section 4 defines the ℓ-LRPC codes and analyzes decoding failure probability and error-correcting capability. In Sect. 5, we apply the ideal ℓ-RD problem and the ideal ℓ-LRPC codes to improve RQC and ROLLO. We conclude this paper in Sect. 6.

2 Notations

- We denote by $\mathbb{N}$ the set of positive integer numbers, q prime or prime power, and $\mathbb{F}_{q^m}$ an extension of degree m of the finite field $\mathbb{F}_q$.
- Let $\alpha \in \mathbb{F}_{q^m}$ be a primitive element and $\boldsymbol{\alpha} = (1, \alpha, \ldots, \alpha^{m-1})$ be a basis of $\mathbb{F}_{q^m}$ viewed as an $\mathbb{F}_q$ vector space.
- Vectors (resp. matrices) are represented by lower-case (resp. upper-case) bold letters. We say that an algorithm is a PPT algorithm if it is a probabilistic polynomial-time algorithm.
- If $\mathcal{X}$ is a finite set, $x \xleftarrow{\$} \mathcal{X}$ (resp. $x \xleftarrow{\mathsf{seed}} \mathcal{X}$) denotes that x is chosen uniformly and randomly from the set $\mathcal{X}$ (resp. by a seed seed).
- For integers $a \leq b$, let $\{a..b\}$ denote all integers from a to b.
- The number of $\mathbb{F}_q$-subspaces of dimension r of $\mathbb{F}_{q^m}$ is given by the Gaussian coefficient $\begin{bmatrix} m \\ r \end{bmatrix}_q = \prod_{i=0}^{r-1} \frac{q^m - q^i}{q^r - q^i} \approx q^{r(m-r)}$.
- The submatrix of a matrix $\boldsymbol{M}$ formed from the rows in I and columns in J is denoted by $\boldsymbol{M}_{I,J}$. When I (resp. J) consists of all the rows (resp. columns), we use the notation $\boldsymbol{M}_{*,J}$ (resp. $\boldsymbol{M}_{I,*}$).
- $|\boldsymbol{M}|$, $|\boldsymbol{M}|_{I,J}$, and $|\boldsymbol{M}|_{*,J}$ are the determinant of the matrix $\boldsymbol{M}$, the submatrix $\boldsymbol{M}_{I,J}$, and the submatrix $\boldsymbol{M}_{*,J}$, respectively.
- $\mathrm{GL}_\eta(\mathbb{F}_q)$ is a general linear group and represents the set of all invertible matrices of size η over $\mathbb{F}_q$. The matrix $\boldsymbol{I}_r$ is the identity matrix of size r.
- The maximal minor c_T of a matrix $\boldsymbol{C}$ of size $r \times n$ is the determinant of its submatrix $\boldsymbol{C}_{*,T}$ whose column indexes $T \subset \{1..n\}$ and $\#T = r$.
- Cauchy-Binet formula that computes the determinant of the product of $\boldsymbol{A} \in \mathbb{F}_{q^m}^{r \times n}$ and $\boldsymbol{B} \in \mathbb{F}_{q^m}^{n \times r}$ is expressed as $|\boldsymbol{AB}| = \sum_{T \subset \{1..n\}, \#T = r} |\boldsymbol{A}|_{*,T} |\boldsymbol{B}|_{T,*}$.
- The Gaussian elimination of a $\mu \times \nu$ matrix of rank ρ over an $\mathbb{F}_q$ has a complexity of $\mathcal{O}(\rho^{\omega-2}\mu\nu)$ operations in $\mathbb{F}_q$, where ω is the exponent of matrix multiplication with $2 \leq \omega \leq 3$ and a practical value is 2.81 when more than a few hundreds rows and columns.
- The complexities are estimated by operations in $\mathbb{F}_q$ if there is no ambiguity. All logarithms are of base 2.

3 The ℓ-RD Problem and Its Complexity

In this section, we first introduce the blockwise errors (ℓ-errors) and the blockwise RD (ℓ-RD) problem in Subsect. 3.1. Then, to analyze the complexity of the ℓ-RD

problem, we refine a universal reduction from existing attacks on the RD problem and analyze the support and coefficient matrices of the ℓ-error in Subsect. 3.2. Finally, we adapt the typical combinatorial and algebraic attacks to the ℓ-RD problem in Subsects. 3.3, 3.4 and 3.5, and find that the ℓ-errors do not ease the problem too much: the ℓ-RD problem is still exponentially hard for appropriate choices of $\ell > 1$.

3.1 The ℓ-Errors and ℓ-RD Problem

Let $\ell, k \in \mathbb{N}$. Let $\boldsymbol{n} = (n_1, \ldots, n_\ell)$ and $\boldsymbol{r} = (r_1, \ldots, r_\ell)$ be vectors of positive integers. Let $n = \sum_{i=1}^{\ell} n_i$ and $r = \sum_{i=1}^{\ell} r_i$. We first define the disjointness of multiple subspaces. We say that ℓ $\mathbb{F}_q$-subspaces $\{V_i\}_{i \in \{1..\ell\}}$ of $\mathbb{F}_{q^m}$ are mutually *disjoint* if $\forall\ i, j \in \{1..\ell\}, i \neq j, V_i \cap V_j = \{0\}$.

Definition 3.1 (Blockwise Errors (ℓ-errors)). *Let $\boldsymbol{e}_i \in \mathbb{F}_{q^m}^{n_i}$ be a vector of weight r_i for $i \in \{1..\ell\}$. An error $\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_\ell) \in \mathbb{F}_{q^m}^{n}$ is called an ℓ-error if the supports of ℓ vectors $\boldsymbol{e}_i$'s are mutually disjoint.*

Recall that $\boldsymbol{n} = (n_1, \ldots, n_\ell)$ and $\boldsymbol{r} = (r_1, \ldots, r_\ell)$ are two vectors of positive integers. We denote the set of such ℓ-errors by $\mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$. Let E_i be the support of dimension r_i of $\boldsymbol{e}_i$. Because all supports are mutually disjoint, the ℓ-error $\boldsymbol{e}$ can be viewed as the error of weight r and support $E = \sum_{i=1}^{\ell} E_i$.

We now define the ℓ-RD problem. This problem is the Rank Decoding (RD) problem finding the ℓ-errors.

Definition 3.2 (Blockwise RD (ℓ-RD) Problem). *Let $\boldsymbol{G}$ be the generator matrix of a random $[n, k]_{q^m}$-linear code $\mathcal{C}$ and $\boldsymbol{y} \in \mathbb{F}_{q^m}^{n}$. The problem is to find $\boldsymbol{x} \in \mathbb{F}_{q^m}^{k}$ and $\boldsymbol{e} \in \mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ such that $\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e}$.*

Like the dual version of the RD problem using the generator matrix is the Rank Syndrome Decoding (RSD) problem [23] using the parity-check matrix, the dual version of the ℓ-RD problem is defined as the ℓ-RSD problem.

Definition 3.3 (Blockwise RSD (ℓ-RSD) Problem). *Let $\boldsymbol{H}$ be the parity-check matrix of a random $[n, k]_{q^m}$-linear code $\mathcal{C}$ and $\boldsymbol{s} \in \mathbb{F}_{q^m}^{n-k}$. The problem is to find $\boldsymbol{e} \in \mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ such that $\boldsymbol{s} = \boldsymbol{H}\boldsymbol{e}^\top$.*

Two variants are exactly the standard RD and RSD problems when $\ell = 1$. By the duality, the hardness of two variants is equivalent. Intuitively, two variants are also hard because they still find a small-weight error.

3.2 Reduction, Support and Coefficient Matrices

In this subsection, we first recall existing attacks on the RD problem, then adapt the reduction refined from typical attacks to the ℓ-RD problem, finally analyze support and coefficient matrices of the ℓ-error.

Attacks on the RD Problem. There currently exist the combinatorial and algebraic attacks [5,7–9,14,21,37] on the RD problem. Please see Appendix B of full version [40] for detailed overviews of these attacks. The first combinatorial attack [14] starts with the RSD problem and is significantly improved in [37] and further refined in [5,21]. The combinatorial attacks [5,14,21] consist of subtly guessing the support of error and solving a linear system. The attack [37] transforms a quadratic multivariate system obtained from the RD problem into a linear system by guessing the entries of support matrix and coefficient matrix. Another way is the algebraic attack [21], where one solves a multivariate system induced from the RD problem based on the annulator polynomial by linearization and Gröbner basis. A breakthrough paper [7] shows that the $\mathbb{F}_{q^m}$-linearity allows to devise a dedicated algebraic attack, i.e., the MaxMinors (MM) modeling. Then the MM modeling is refined and improved in [9] where the authors also introduced another algebraic modeling, the Support-Minors (SM) modeling. The SM modeling later is combined with the MM modeling (i.e., the SM-$\mathbb{F}_{q^m}^{+}$ modeling [8]). Both SM and MM modelings reduce the RD problem to solving a linear system. The analysis in [8] shows that the cost of the SM-$\mathbb{F}_{q^m}^{+}$ modeling is close to those of the combinatorial attack [5] and the MM modeling [9].

To measure the potential complexity loss and ensure the security of schemes, we adapt typical combinatorial attacks [5,37] and algebraic attacks [9,21] to the ℓ-RD problem in Subsects. 3.3, 3.4 and 3.5. The reduction technique in attacks [5,9,21,37] is still available to the ℓ-RD problem. We refine the reduction in Theorem 3.4.

Theorem 3.4. *Solving the ℓ-RD(q,m,n,k,r,ℓ) problem defined by $[n,k]_{q^m}$ linear code $\mathcal{C}$ (see Definition 3.2) can be reduced to finding a blockwise codeword (i.e., an ℓ-error) of weight r in the $[n,k+1]_{q^m}$ extended code of $\mathcal{C}$.*

Proof. Once obtaining word $\boldsymbol{y}$, one adds $\boldsymbol{y}$ to code $\mathcal{C}$ and obtains an $[n,k+1]_{q^m}$ extended code $\mathcal{C}_{\boldsymbol{y}} = \mathcal{C} + \langle \boldsymbol{y} \rangle$ with a generator matrix $\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{G} \end{pmatrix}$ of size $(k+1) \times n$. In this way, $\boldsymbol{e} = \begin{pmatrix} 1 & -\boldsymbol{m} \end{pmatrix} \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{G} \end{pmatrix}$ is exactly a codeword of weight r of $\mathcal{C}_{\boldsymbol{y}}$. Let $\boldsymbol{G}_{\boldsymbol{y}} = (\boldsymbol{I}_{k+1} \ \boldsymbol{R}) \in \mathbb{F}_{q^m}^{(k+1)\times n}$ be a systematic generator matrix of $\mathcal{C}_{\boldsymbol{y}}$ and $\boldsymbol{H}_{\boldsymbol{y}} = \begin{pmatrix} -\boldsymbol{R}^{\top} & \boldsymbol{I}_{n-k-1} \end{pmatrix} \in \mathbb{F}_{q^m}^{(n-k-1)\times n}$ be a systematic parity-check matrix of $\mathcal{C}_{\boldsymbol{y}}$, where $\boldsymbol{R} \in \mathbb{F}_{q^m}^{(k+1)\times(n-k-1)}$. Then solving the ℓ-RD problem consists in finding an $\boldsymbol{u} \in \mathbb{F}_{q^m}^{k+1}$ of weight $\leq r$ such that

$$\boldsymbol{u}\boldsymbol{G}_{\boldsymbol{y}} = \boldsymbol{e}, \tag{2}$$

or finding an ℓ-error $\boldsymbol{e}$ of weight r such that

$$\boldsymbol{e}\boldsymbol{H}_{\boldsymbol{y}}^{\top} = \boldsymbol{0}. \tag{3}$$

□

The support and coefficient matrices of the ℓ-error are crucial tools to construct the specific attack modelings by exploiting the reduction in Theorem 3.4. The entries of two matrices determine the number of variables of algebraic equations in the attack modelings. We next analyze the forms of two matrices.

Support and Coefficient Matrices of the ℓ-error. Let $\boldsymbol{n} = (n_1, \ldots, n_\ell)$ and $\boldsymbol{r} = (r_1, \ldots, r_\ell)$ be vectors of positive integers. Let $\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_\ell) \in \mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ be an ℓ-error. If let $\boldsymbol{\varepsilon}_i = (\varepsilon_{i1}, \varepsilon_{i2}, \ldots, \varepsilon_{ir_i}) \in \mathbb{F}_{q^m}^{r_i}$ be a basis of support of dimension r_i, then there exists a matrix $\boldsymbol{C}_i \in \mathbb{F}_q^{r_i \times n_i}$ of rank r_i such that $\boldsymbol{e}_i = \boldsymbol{\varepsilon}_i \boldsymbol{C}_i$, If one expresses the basis $\boldsymbol{\varepsilon}_i$ as a matrix $\boldsymbol{S}_i \in \mathbb{F}_q^{m \times r_i}$ of rank r_i under the basis $\boldsymbol{\alpha}$, then $\boldsymbol{e}_i = \boldsymbol{\alpha} \boldsymbol{S}_i \boldsymbol{C}_i$. We have $\boldsymbol{e} = \boldsymbol{\varepsilon} \boldsymbol{C} = \boldsymbol{\alpha} \boldsymbol{S} \boldsymbol{C}$, where $\boldsymbol{\varepsilon} = (\boldsymbol{\varepsilon}_1, \boldsymbol{\varepsilon}_2, \ldots, \boldsymbol{\varepsilon}_\ell) \in \mathbb{F}_{q^m}^{r}$,

$$\boldsymbol{S} = \begin{pmatrix} \boldsymbol{S}_1 & \boldsymbol{S}_2 & \cdots & \boldsymbol{S}_\ell \end{pmatrix} \in \mathbb{F}_q^{m \times r}, \qquad \boldsymbol{C} = \begin{pmatrix} \boldsymbol{C}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_2 & \boldsymbol{0} & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{C}_\ell \end{pmatrix} \in \mathbb{F}_q^{r \times n}. \tag{4}$$

We call $\boldsymbol{S}$ and $\boldsymbol{C}$ respectively support matrix and coefficient matrix of $\boldsymbol{e}$.

Remark 1. The main difference with the standard rank metric error is that the form of the coefficient matrix $\boldsymbol{C}$ of the ℓ-error is of block-diagonal form. For a standard rank metric error $\boldsymbol{e} \in \mathcal{S}_r^n$, let $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_r) \in \mathbb{F}_{q^m}^r$ be a basis of $\mathrm{Supp}(\boldsymbol{e})$, the there is a coefficient matrix $\boldsymbol{C} \in \mathbb{F}_q^{r \times n}$ of rank r such that $\boldsymbol{e} = \boldsymbol{\varepsilon} \boldsymbol{C}$. Under the basis $\boldsymbol{\alpha}$, there is a support matrix $\boldsymbol{S} \in \mathbb{F}_q^{m \times r}$ of rank r such that $\boldsymbol{\varepsilon} = \boldsymbol{\alpha} \boldsymbol{S}$. Then $\boldsymbol{e} = \boldsymbol{\alpha} \boldsymbol{S} \boldsymbol{C}$.

Support and Coefficient Matrices with Less Entries. Because all multiples $\lambda \boldsymbol{e}$ for $\lambda \in \mathbb{F}_{q^m}^*$ are solutions of Eq. (3) due to $\|\lambda \boldsymbol{e}\|_{\mathrm{R}} = r$, one can specify λ to be the inverse of the first coordinate of $\boldsymbol{e}$. Without loss of generality, let the first coordinate of $\boldsymbol{e}$ be 1, then one can set the first column of $\boldsymbol{C}$ to $(1\ 0\ \cdots\ 0)^\top$ and the first column of $\boldsymbol{S}$ to $(1\ 0\ \cdots\ 0)^\top$. Then $\boldsymbol{S}$ and $\boldsymbol{C}$ can be further reduced to two forms with less entries.

- $\boldsymbol{S}_{\{1..r\},*} = \boldsymbol{I}_r$. By Gaussian elimination on column of $\boldsymbol{S}$, there is a matrix $\boldsymbol{P} \in \mathrm{GL}_r(q)$ such that $\boldsymbol{S}\boldsymbol{P} = \left(\begin{array}{c} \boldsymbol{I}_r \\ \hline \boldsymbol{0}_{(m-r)\times 1} \end{array} \middle| \boldsymbol{S}'\right)$ and $\boldsymbol{P}^{-1}\boldsymbol{C} = \left(\begin{array}{c|c} 1 & \\ \boldsymbol{0}_{(r-1)\times 1} & \boldsymbol{C}' \end{array}\right)$ where $\boldsymbol{S}' \in \mathbb{F}_q^{(m-r)\times(r-1)}$ and $\boldsymbol{C}' \in \mathbb{F}_q^{r\times(n-1)}$. Then

$$\boldsymbol{e} = \boldsymbol{\alpha}\boldsymbol{S}\boldsymbol{C} = \boldsymbol{\alpha}\boldsymbol{S}\boldsymbol{P}\boldsymbol{P}^{-1}\boldsymbol{C} = \boldsymbol{\alpha}\left(\begin{array}{c} \boldsymbol{I}_r \\ \hline \boldsymbol{0}_{(m-r)\times 1} \end{array} \middle| \boldsymbol{S}'\right)\left(\begin{array}{c|c} 1 & \\ \boldsymbol{0}_{(r-1)\times 1} & \boldsymbol{C}' \end{array}\right). \tag{5}$$

  Let $\boldsymbol{s} := \boldsymbol{S}\boldsymbol{P}$ and $\boldsymbol{C} := \boldsymbol{P}^{-1}\boldsymbol{C}$.
- $\boldsymbol{C}_i$ is of systematic form. By Gaussian elimination on row of $\boldsymbol{C}$, there is a matrix $\boldsymbol{Q}_i \in \mathrm{GL}_{r_i}(q)$ such that $\boldsymbol{Q}_i\boldsymbol{C}_i = (\boldsymbol{I}_{r_i}\ \ \boldsymbol{C}_i')$ and $\boldsymbol{S}\boldsymbol{Q}^{-1} = \left(\begin{array}{c|c} 1 & \\ \boldsymbol{0}_{(m-1)\times 1} & \boldsymbol{S}' \end{array}\right)$ where $\boldsymbol{C}_i' \in \mathbb{F}_q^{r_i\times(n_i-r_i)}$, $\boldsymbol{S}' \in \mathbb{F}_q^{m\times(r-1)}$, and $\boldsymbol{Q} =$

$\begin{pmatrix} \boldsymbol{Q}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_2 & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{Q}_\ell \end{pmatrix} \in \mathrm{GL}_r(q)$. Then

$$\boldsymbol{e} = \boldsymbol{\alpha}\boldsymbol{S}\boldsymbol{C} = \boldsymbol{\alpha}\boldsymbol{S}\boldsymbol{Q}^{-1}\boldsymbol{Q}\boldsymbol{C} = \boldsymbol{\alpha}\left(\begin{array}{c|c} 1 & \\ \mathbf{0}_{(m-1)\times 1} & \boldsymbol{S}' \end{array}\right)\begin{pmatrix} \boldsymbol{Q}_1\boldsymbol{C}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_2\boldsymbol{C}_2 & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{Q}_\ell\boldsymbol{C}_\ell \end{pmatrix}. \tag{6}$$

Let $\boldsymbol{S} := \boldsymbol{S}\boldsymbol{Q}^{-1}$ and $\boldsymbol{C} := \boldsymbol{Q}\boldsymbol{C}$.

For solving the ℓ-RD problem, most attacks aim to recover $\boldsymbol{S}$ and $\boldsymbol{C}$ by solving the algebraic equations obtained from Eqs. (2)–(6). Equation (3) is used to build the AGHT attacks (Subsect. 3.3). Equations (2), (5) and (6) are used to build the OJ attack (Subsect. 3.3). Equations (3) and (6) are used to build the algebraic attack, the MM modeling (Subsect. 3.5). The details of constructing the algebraic equations can refer to the specific attacks in Subsects. 3.3, 3.4 and 3.5.

3.3 Combinatorial Attacks on the ℓ-RD Problem

In this subsection, we use the AGHT attack [5] and the OJ attack [37] to analyze the complexity of solving the ℓ-RD problem.

AGHT Attack [5]**.** The idea is that the solver tries to guess a subspace that contains the support of the ℓ-error, then checks if the choice is correct. The cost depends on how to successfully guess such a subspace.

- Guess randomly a t-dimensional subspace F that contains the support $\mathrm{Supp}(\boldsymbol{e})$ of dimension $r = \sum_{i=1}^{\ell} r_i$ of the ℓ-error $\boldsymbol{e}$.
- Let $(f_1, f_2, \ldots, f_t) \in \mathbb{F}_{q^m}^t$ be a basis of F. One expresses $\boldsymbol{e}$ under this basis

$$\boldsymbol{e} = (e_1, e_2, \ldots, e_n) = (f_1, f_2, \ldots, f_t)\begin{pmatrix} e_{11} & e_{12} & \cdots & e_{1n} \\ e_{21} & e_{22} & \cdots & e_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ e_{t1} & e_{t2} & \cdots & e_{tn} \end{pmatrix} = (f_1, f_2, \ldots, f_t)\begin{pmatrix} \overline{\boldsymbol{e}}_1 \\ \overline{\boldsymbol{e}}_2 \\ \vdots \\ \overline{\boldsymbol{e}}_t \end{pmatrix},$$

where $\overline{\boldsymbol{e}}_i = (e_{i1}, e_{i2}, \ldots, e_{in}) \in \mathbb{F}_q^n$ for $i \in \{1..t\}$. By Eq. (3): $\boldsymbol{H_y e}^\top = \boldsymbol{0}$, let $\boldsymbol{h}_j$ is the j-th row of $\boldsymbol{H_y}$, we have

$$\begin{aligned}\boldsymbol{H_y e}^\top &= \begin{pmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \vdots \\ \boldsymbol{h}_{n-k-1} \end{pmatrix} \left(\overline{\boldsymbol{e}}_1^\top, \overline{\boldsymbol{e}}_2^\top, \ldots, \overline{\boldsymbol{e}}_t^\top\right) \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_t \end{pmatrix} \\ &= \begin{pmatrix} \boldsymbol{h}_1 f_1 & \boldsymbol{h}_1 f_2 & \cdots & \boldsymbol{h}_1 f_t \\ \boldsymbol{h}_2 f_1 & \boldsymbol{h}_2 f_2 & \cdots & \boldsymbol{h}_2 f_t \\ \vdots & \vdots & \cdots & \vdots \\ \boldsymbol{h}_{n-k-1} f_1 & \boldsymbol{h}_{n-k-1} f_2 & \cdots & \boldsymbol{h}_{n-k-1} f_t \end{pmatrix} \begin{pmatrix} \overline{\boldsymbol{e}}_1^\top \\ \overline{\boldsymbol{e}}_2^\top \\ \vdots \\ \overline{\boldsymbol{e}}_t^\top \end{pmatrix} = \boldsymbol{0}_{n-k-1}. \end{aligned} \tag{7}$$

- Express Eq. (7) as a linear system over $\mathbb{F}_q$ and solve $\overline{\boldsymbol{e}}_i$. By expressing $\boldsymbol{h}_j f_i$ as a matrix $\mathrm{Mat}(\boldsymbol{h}_j f_i) \in \mathbb{F}_q^{m \times n}$ under the basis $\boldsymbol{\alpha}$ for $j \in \{1..n-k-1\}$ and $i \in \{1..t\}$, a linear system over $\mathbb{F}_q$ with nt unknowns and $m(n-k-1)$ equations is obtained. The linear system has only one solution with overwhelming probability if $nt \leq m(n-k-1)$.
- The probability of $F \supset E$ is estimated as $\frac{\begin{bmatrix} t \\ r \end{bmatrix}_q}{\begin{bmatrix} m \\ r \end{bmatrix}_q} \approx q^{-r(m-t)}$. In this way, the complexity is $\mathcal{O}\left((n-k-1)m)^\omega q^{r\left\lceil \frac{(k+1)m}{n} \right\rceil}\right)$.
- Use $\mathbb{F}_{q^m}$-linearity to decrease the cost. Since, for any $\lambda \in \mathbb{F}_{q^m}^*$, $\|\lambda \boldsymbol{e}\|_{\mathrm{R}} = r$ and all multiples $\lambda \boldsymbol{e}$ are solutions of Eq. (3): $\boldsymbol{H_y e}^\top = \boldsymbol{0}$, the complexity is divided by about q^m.

As a result, this attack has a complexity of $\mathcal{O}\left((n-k-1)m)^\omega q^{r\left\lceil \frac{(k+1)m}{n} \right\rceil - m}\right)$.

In [12], the authors adapted the AGHT attack to the RD problem finding so-called non-homogeneous errors. Here, inspired by [12], the strategy guessing the subspace F is that the solver randomly guesses a subspace F_i of dimension t_i that contains the support $E_i = \mathrm{Supp}(\boldsymbol{e}_i)$ of dimension r_i of $\boldsymbol{e}_i$ such that all F_i's are mutually disjoint, and sets $F = \sum_{i=1}^{\ell} F_i$. In this way, the dimension of F is of $\sum_{i=1}^{\ell} t_i$, and F must contain the support of the ℓ-error $\boldsymbol{e}$.

If one knows F_i, then each entry of $\boldsymbol{e}_i$ can be expressed as an $\mathbb{F}_q$-linear combination of t_i elements in a basis of F_i. This means that one can write $\boldsymbol{e}_i$ using $n_i t_i$ unknowns in $\mathbb{F}_q$. Doing the same for all $\boldsymbol{e}_i$'s, one obtains $\sum_{i=1}^{\ell} n_i t_i$ unknowns. Then one solves the linear system with $\sum_{i=1}^{\ell} n_i t_i$ unknowns and $m(n-k-1)$ equations for single solution $\boldsymbol{e}$ as long as $\sum_{i=1}^{\ell} n_i t_i \leq m(n-k-1)$. The most costly part of the attack consists in finding the F_i's containing E_i for $i \in \{1..\ell\}$. We estimate this probability in Lemma 3.5.

Lemma 3.5. *Let $E_1, E_2, \ldots, E_\ell$ be fixed $\mathbb{F}_q$-subspaces of dimension respectively $r_1, r_2, ..., r_\ell$ of $\mathbb{F}_{q^m}$. The probability that one successfully guesses $\mathbb{F}_q$-subspaces $F_1, F_2, ..., F_\ell$ dimension respectively $t_1, t_2, \ldots, t_\ell$ of $\mathbb{F}_{q^m}$ such that all F_i's are mutually disjoint and $E_i \subset F_i$ is estimated as $\mathcal{O}\left(q^{-mr + \sum_{i=1}^{\ell-1} r_i^2 + \sum_{j=2}^{\ell} r_j \sum_{i=1}^{j-1} r_i + t_\ell r_\ell}\right)$.*

We give the detailed proof for Lemma 3.5 in Appendix C.1 of full version [40]. Finally, one takes advantage of the $\mathbb{F}_{q^m}$-linearity to raise this probability: for any $\lambda \in \mathbb{F}_{q^m}^*$, $\|\lambda \boldsymbol{e}\|_{\mathrm{R}} = r$ and all multiples $\lambda \boldsymbol{e}$ are solutions of Eq. (3): $\boldsymbol{H_y} \boldsymbol{e}^\top = \boldsymbol{0}$, hence the complexity is divided by about q^m. The complexity of solving the ℓ-RD problem by the variant of AGHT attack is estimated as

$$\mathcal{O}\left((m(n-k-1))^\omega q^{mr-\sum_{i=1}^{\ell-1} r_i^2 - \sum_{j=2}^{\ell} r_j \sum_{i=1}^{j-1} r_i - t_\ell r_\ell - m}\right)$$

where t_i is chosen to maximize $t_\ell r_\ell$ under the constraints

$$\begin{cases} r_i \le t_i, \quad for\ i \in \{1..\ell\}; \\ \sum_{i=1}^{\ell} t_i \le m-1; \\ \sum_{i=1}^{\ell} n_i t_i \le m(n-k-1). \end{cases}$$

OJ Attack. We now analyze the complexity of solving the ℓ-RD problem by the OJ attack [37]. Let $\overline{\boldsymbol{e}}_1$ and $\overline{\boldsymbol{e}}_2$ be the first $k+1$ and the last $n-k-1$ coordinates of $\boldsymbol{e}$. Let $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ be the first $k+1$ columns and the last $n-k-1$ columns of $\boldsymbol{C}$. Then $\boldsymbol{e} = (\overline{\boldsymbol{e}}_1, \overline{\boldsymbol{e}}_2) = \boldsymbol{\varepsilon}(\boldsymbol{A}_1, \boldsymbol{A}_2) = (\boldsymbol{\alpha S A}_1, \boldsymbol{\alpha S A}_2)$. Equation (2) means

$$\boldsymbol{u G_y} = \boldsymbol{e} \Longleftrightarrow (\boldsymbol{u}\ \boldsymbol{uR}) = (\overline{\boldsymbol{e}}_1, \overline{\boldsymbol{e}}_2) \Longleftrightarrow \overline{\boldsymbol{e}}_1 \boldsymbol{R} = \overline{\boldsymbol{e}}_2 \Longleftrightarrow \boldsymbol{\alpha S A}_1 \boldsymbol{R} = \boldsymbol{\alpha S A}_2. \quad (8)$$

We first analyze the case of the 2-RD problem, then extend conclusions into general cases. By Equation (8), for $j \in \{1..n-k-1\}$, let $\boldsymbol{r}_j$ and $\boldsymbol{a}_j$ be the j-th column of $\boldsymbol{R}$ and $\boldsymbol{A}_2$, respectively, then

$$\boldsymbol{\alpha S A}_1 \boldsymbol{r}_j = \boldsymbol{\alpha S a}_j \iff \boldsymbol{\alpha S} \begin{pmatrix} \boldsymbol{A}_1 & \boldsymbol{a}_j \end{pmatrix} \begin{pmatrix} \boldsymbol{r}_j \\ -1 \end{pmatrix} = 0. \quad (9)$$

Let $\begin{pmatrix} \boldsymbol{r}_j \\ -1 \end{pmatrix} = \boldsymbol{T}_j \boldsymbol{\alpha}^\top$ where $\boldsymbol{T}_j \in \mathbb{F}_q^{(k+2)\times m}$ is the matrix expression of $\begin{pmatrix} \boldsymbol{r}_j \\ -1 \end{pmatrix}$ under the basis $\boldsymbol{\alpha}$. Equation (9) can be written $\boldsymbol{\alpha S} \begin{pmatrix} \boldsymbol{A}_1 & \boldsymbol{a}_j \end{pmatrix} \boldsymbol{T}_j \boldsymbol{\alpha}^\top = 0$. This means

$$\boldsymbol{S}(\boldsymbol{A}_1\ \boldsymbol{a}_j)\boldsymbol{T}_j = \boldsymbol{0}_{m\times m}. \quad (10)$$

The entries of $\boldsymbol{S}(\boldsymbol{A}_1\ \boldsymbol{a}_j)\boldsymbol{T}_j$ are quadratic polynomials. Then Eq. (10) gives a quadratic multivariate system over $\mathbb{F}_q$ with m^2 quadratic polynomials in the entries of $\boldsymbol{S}$ and $\boldsymbol{C}$.

The OJ attack uses the basis enumeration and the coordinates enumeration to transform the quadratic multivariate system into a linear system. The former guesses all entries of $\boldsymbol{S}$ and solves the linear system about the entries of $(\boldsymbol{A}_1\ \boldsymbol{a}_j)$ to determine $\boldsymbol{C}$. The latter guesses the entries of $\boldsymbol{C}$ and solves the linear system about the entries of $\boldsymbol{S}$ to determine $\boldsymbol{S}$.

When $\boldsymbol{S}$ and $\boldsymbol{C}$ are in the form of Eq. (5) and Eq. (6), the complexities are presented in Theorem 3.6 and Theorem 3.7. We give their detailed proofs in Appendix C.2 and Appendix C.3 of full version [40]. The ideas of proofs can be easily extended to the ℓ-RD problem.

Theorem 3.6. *If $\boldsymbol{S}$ and $\boldsymbol{C}$ are in the form of Eq. (5), the 2-RD problem can be solved with complexity $\mathcal{O}\left((kr+r)^{\omega} q^{(m-r)(r-1)}\right)$ by the basis enumeration.*

Theorem 3.7. *If $k = n_1$, $\boldsymbol{S}$ and $\boldsymbol{C}$ are in the form of Eq. (6), the 2-RD problem can be solved with complexity $\mathcal{O}\left((m(r-1)+(n_1-r_1))^{\omega} q^{(r_1-1)(n_1-r_1)+r_2}\right)$ by the coordinates enumeration.*

Theorem 3.8. *If $k = n_1$, the complexity of solving the ℓ-RD problem by the OJ attack is estimated as*

$$\begin{cases} \mathcal{O}\left((kr+r)^{\omega} q^{(m-r)(r-1)}\right), & \textit{Basis Enumeration}; \\ \mathcal{O}\left((m(r-1)+(n_1-r_1))^{\omega} q^{(r_1-1)(n_1-r_1)+\gamma}\right), & \textit{Coordinates Enumeration}, \end{cases}$$

where $\gamma = \max\left\{r_i : i \in \{2..\ell\}\right\}$ and $r = \sum_{i=1}^{\ell} r_i$.

3.4 Algebraic Attack by Annulator Polynomial

This algebraic attack [21] differs from attacks aiming to recover $\boldsymbol{S}$ and $\boldsymbol{C}$ with reductions described in Subsect. 3.2. It directly solves $\boldsymbol{x}$ from a multivariate system obtained from the ℓ-RD instance and the theory of q-polynomials [36], more specifically annulator polynomials (see Appendix A of full version [40]). The attack details are outlined in Appendix B.2 of full version [40].

For the ℓ-RD problem finding the ℓ-error $\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_\ell) \in \mathcal{S}_r^{\boldsymbol{n}}$, the solver splits $\boldsymbol{y}$ as $(\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_\ell)$ and splits $\boldsymbol{G}$ as $(\boldsymbol{G}_1, \boldsymbol{G}_2, \ldots, \boldsymbol{G}_\ell)$ by columns $\boldsymbol{n}$. Then

$$(\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_\ell) = \boldsymbol{x}(\boldsymbol{G}_1, \boldsymbol{G}_2, \ldots, \boldsymbol{G}_\ell) + (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_\ell).$$

In this way, the ℓ-RD problem is divided into ℓ subproblems, for $\nu \in \{1..\ell\}$, $\boldsymbol{y}_\nu = \boldsymbol{x}\boldsymbol{G}_\nu + \boldsymbol{e}_\nu$, then one solves $\boldsymbol{x}$ from *one* of ℓ subproblems.

Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_k)$. For $\nu \in \{1..\ell\}$, let $\boldsymbol{y}_\nu = (y_1, y_2, \ldots, y_{n_\nu})$, $\boldsymbol{G}_\nu = (g_{ij})_{\substack{i\in\{1..k\} \\ j\in\{1..n_\nu\}}}$, and $\boldsymbol{e}_\nu = (e_1, e_2, \ldots, e_{n_\nu})$. Since the entries of $\boldsymbol{e}_\nu$ lie in the support $\mathrm{Supp}(\boldsymbol{e}_\nu)$ of dimension r_ν, there exists a unique monic q-polynomials $P^{(\nu)}(u) = \sum_{\delta=0}^{r_\nu} p_\delta^{(\nu)} u^{q^\delta}$ of q-degree r_ν such that for $j \in \{1..n_\nu\}$

$$P^{(\nu)}\left(y_j - \sum_{i=1}^{k} x_i g_{ij}\right) = \sum_{\delta=0}^{r_\nu}\left(p_\delta^{(\nu)} y_j^{q^\delta} - \sum_{i=1}^{k} p_\delta^{(\nu)} x_i^{q^\delta} g_{ij}^{q^\delta}\right) = P^{(\nu)}(e_j) = 0. \quad (11)$$

Equation (11) gives a multivariate system with n_ν polynomials and $(r_\nu + k)$ variables $p_\delta^{(\nu)}$ and x_i. For solving the ℓ-RD problem, one solves x_i from this multivariate system.

The linearization and Gröbner basis techniques are applied to solve x_i. The complexities are given in Theorem 3.9 and the detailed proof is presented in Appendix C.4 of full version [40].

Theorem 3.9. *The complexity of solving the ℓ-RD problem by annulator polynomials is estimated as*

$$\begin{cases} \mathcal{O}\left(\min\left\{(r_\nu k)^\omega q^{r_\nu\left\lceil\frac{(k+1)(r_\nu+1)-(n_\nu+1)}{r_\nu}\right\rceil} : \nu \in \{1..\ell\}\right\}\right), & \textit{Linearization;} \\ \mathcal{O}\left(\min\left\{n_\nu \binom{r_\nu+k+d_{reg}^{(\nu)}-1}{d_{reg}^{(\nu)}}^\omega : \nu \in \{1..\ell\}\right\}\right), & \textit{Gröbner Basis.} \end{cases}$$

where $d_{reg}^{(\nu)}$ is the degree of regularity of the semi-regular system.

3.5 Algebraic Attacks by the MaxMinors Modeling

The MaxMinors (MM) modeling [9] is a powerful algebraic attack for cryptographic parameters and reduces the RD problem to solving a linear system. Equation $\boldsymbol{\varepsilon C H}_{\boldsymbol{y}}^\top = \mathbf{0}_{n-k-1}$ (obtained from Eq. (3) and $\boldsymbol{e} = \boldsymbol{\varepsilon C}$) implies that $\boldsymbol{C H}_{\boldsymbol{y}}^\top \in \mathbb{F}_{q^m}^{r\times(n-k-1)}$ is not of row full rank because a non-zero vector $\boldsymbol{s}$ belongs to its left kernel. Then all maximal minors $|\boldsymbol{C H}_{\boldsymbol{y}}^\top|_{*,J}$ of $\boldsymbol{C H}_{\boldsymbol{y}}^\top$ are equal to 0 for $J \subset \{1..n-k-1\}$ and $\#J = r$. By the Cauchy-Binet formula, each $|\boldsymbol{C H}_{\boldsymbol{y}}^\top|_{*,J}$ can be viewed a non-zero linear combination about all maximal minors $c_T = |\boldsymbol{C}|_{*,T}$ for $T \subset \{1..n\}$ and $\#T = r$. One views non-zero c_T as unknowns and solves c_T from a linear system with $\binom{n}{r}$ unknowns and $\binom{n-k-1}{r}$ equations. Finally, one determines the entries of $\boldsymbol{C}$ from the c_T by using the fact that it is in systematic form. The MM modeling over $\mathbb{F}_{q^m}$ is built

$$\left\{P_J = |\boldsymbol{C H}_{\boldsymbol{y}}^\top|_{*,J} : J \subset \{1..n-k-1\}, \#J = r\right\}, \quad \text{(MM-}\mathbb{F}_{q^m}\text{)} \tag{12}$$

Unknowns: $\binom{n}{r}$ variables $c_T \in \mathbb{F}_q$ for $T \subset \{1..n\}$ and $\#T = r$,
Equations: $\binom{n-k-1}{r}$ linear equations $P_J = 0$ over $\mathbb{F}_{q^m}$ in c_T.

However, this system has many solutions due to $\binom{n-k-1}{r} < \binom{n}{r}$ whereas one wants more equations than unknowns for a unique solution. To obtain more equations than unknowns, one unfolds the coefficients of P_J over $\mathbb{F}_q$ and obtains the MM-$\mathbb{F}_q$ modeling

$$\left\{P_{i,J} = |\boldsymbol{C H}_{\boldsymbol{y}}^\top|_{*,J} : J \subset \{1..n-k-1\}, \#J = r, i \in \{1..m\}\right\}, \quad \text{(MM-}\mathbb{F}_q\text{)} \tag{13}$$

Unknowns: $\binom{n}{r}$ variables $c_T \in \mathbb{F}_q$ for $T \subset \{1..n\}$ and $\#T = r$,
Equations: $m\binom{n-k-1}{r}$ linear equations $P_{i,J} = 0$ over $\mathbb{F}_q$ in c_T.

We first analyze the case of the 2-RD problem, then extend conclusions to general cases. By Eq. (6), the matrix $\boldsymbol{C}$ is of form

$$\boldsymbol{C} = \left(\begin{array}{cc|c} \boldsymbol{I}_{r_1} & \boldsymbol{C}_1' & \mathbf{0}_{r_1\times n_2} \\ \hline \mathbf{0}_{r_2\times n_1} & \boldsymbol{I}_{r_2} & \boldsymbol{C}_2' \end{array}\right) \in \mathbb{F}_q^{r\times n}, \tag{14}$$

where $\boldsymbol{C} = (c_{ij})_{\substack{i\in\{1..r\}\\ j\in\{1..n\}}} \in \mathbb{F}_q^{r\times n}$, $\boldsymbol{C}_1' \in \mathbb{F}_q^{r_1\times(n_1-r_1)}$, and $\boldsymbol{C}_2' \in \mathbb{F}_q^{r_2\times(n_2-r_2)}$. One can easily check

- $|\boldsymbol{C}|_{*,(\{1..r_1\}\setminus\{i\}\cup\{j\})\cup\{n_1+1..n_1+r_2\}} = (-1)^{r_1-i}c_{ij}$ for $i \in \{1..r_1\}$ and $j \in \{r_1+1..n_1\}$,
- $|\boldsymbol{C}|_{*,\{1..r_1\}\cup(\{n_1+1..n_1+r_2\}\setminus\{i\}\cup\{j\})} = (-1)^{n_1+r_2-i}c_{ij}$ for $i \in \{n_1+1..n_1+r_2\}$ and $j \in \{n_1+r_2+1..n\}$,
- $|\boldsymbol{C}|_{*,\{1..r_1\}\cup\{n_1+1..n_1+r_2\}} = 1$.

Therefore, once all c_T's are solved, one can determine the entries of the matrix $\boldsymbol{C}$. Lemma 3.10 bounds the number of equations and unknowns c_T.

Lemma 3.10. *Under block form of $\boldsymbol{C}$ in Eq. (14), the MM-$\mathbb{F}_q$ modeling obtained from the 2-RD problem contains $\binom{n_1}{r_1}\binom{n_2}{r_2}$ unknowns c_T and at most $m\binom{n-k-1}{r}$ equations.*

We give the detailed proof for Lemma 3.10 in Appendix C.5 of full version [40].

Remark 2. Our analysis follows the idea of updated RQC [30], where authors bounded the maximal number of equations. On the one hand, considering less equations could lead to a higher complexity because in this case one is more likely to solve an underdetermined system with more unknowns and would guess more entries of $\boldsymbol{C}$ to transform the system into an overdetermined case (see hybrid method in the proof of Theorem 3.11). This means that using the maximal number of equations would give a lower bound of complexity. Cryptographic parameters often lead to an underdetermined case. On the other hand, the number of zero and dependent equations is negligible to the maximal number $m\binom{n-k-1}{r}$ and their impact on complexity is very limited. A thorough analysis in [8,12] supported this point and we also experimentally verified this when $\ell = 2, 3$.

Remark 3. The number of non-zero variables c_T is easy to compute. When n and r are divisible by ℓ, by Stirling approximation, the loss of variables c_T is large due to $\binom{n/\ell}{r/\ell}^{\ell} \approx \ell^{\frac{\ell}{2}}\left(\frac{n}{2\pi r(n-r)}\right)^{\frac{\ell-1}{2}}\binom{n}{r}$ while comparing with the MM-$\mathbb{F}_q$ modeling obtained from the standard RD problem. See Lemma C.1 in Appendix C.6 of full version [40] for this proof.

Theorem 3.11. *The complexity of solving the 2-RD problem by the MM-$\mathbb{F}_q$ modeling is estimated as*

$$\begin{cases} \mathcal{O}\left(m\binom{n-p-k-1}{r}\left(\binom{n_1}{r_1}\binom{n_2-p}{r_2}\right)^{\omega-1}\right), & m\binom{n-k-1}{r} \geq \binom{n_1}{r_1}\binom{n_2}{r_2} - 1; \\ \mathcal{O}\left(q^{a_1r_1+a_2r_2}m\binom{n-k-1}{r}\left(\binom{n_1-a_1}{r_1}\binom{n_2-a_2}{r_2}\right)^{\omega-1}\right), & m\binom{n-k-1}{r} < \binom{n_1}{r_1}\binom{n_2}{r_2} - 1. \end{cases}$$

where $p = \max\left\{i \mid m\binom{n-i-k-1}{r} \geq \binom{n_1}{r_1}\binom{n_2-i}{r_2} - 1\right\}$ and (a_1, a_2) is an integer pair such that $m\binom{n-k-1}{r} \geq \binom{n_1-a_1}{r_1}\binom{n_2-a_2}{r_2} - 1$ exactly holds.

We give a proof with full details for Theorem 3.11 in Appendix C.7 of full version [40]. Theorem 3.11 can be extended to the case of the ℓ-RD problem.

Theorem 3.12. *The complexity of solving the ℓ-RD problem by the MM-$\mathbb{F}_q$ modeling is estimated as*

$$\begin{cases} \mathcal{O}\left(m\binom{n-p-k-1}{r}\left(\binom{n_\ell-p}{r_\ell}\prod_{i=1}^{\ell-1}\binom{n_i}{r_i}\right)^{\omega-1}\right), & m\binom{n-k-1}{r} \geq \prod_{i=1}^{\ell}\binom{n_i}{r_i} - 1; \\ \mathcal{O}\left(q^{\sum_{i=1}^{\ell} a_i r_i} m\binom{n-k-1}{r}\left(\prod_{i=1}^{\ell}\binom{n_i-a_i}{r_i}\right)^{\omega-1}\right), & m\binom{n-k-1}{r} < \prod_{i=1}^{\ell}\binom{n_i}{r_i} - 1. \end{cases}$$

where $p = \max\left\{i \;\middle|\; m\binom{n-i-k-1}{r} \geq \binom{n_\ell-i}{r_\ell}\prod_{i=1}^{\ell-1}\binom{n_i}{r_i} - 1\right\}$ *and* $(a_1, a_2, \ldots, a_\ell)$ *is an integers sequence such that* $m\binom{n-k-1}{r} \geq \prod_{i=1}^{\ell}\binom{n_i-a_i}{r_i} - 1$ *exactly holds.*

3.6 Summary of Complexities for Solving the ℓ-RD Problem

At the end of this section, we summarize the complexity gain of solving the ℓ-RD problem compared with the standard RD problem in Table 2. For the first three attacks, we only compare the exponential terms.

Table 2. Complexity comparisons of solving the ℓ-RD and RD problems.

Attacks	RD(q, m, n, k, r)	ℓ-RD(q, m, n, k, r, ℓ)
AGHT	$q^{r\left\lceil\frac{(k+1)m}{n}\right\rceil - m}$	$q^{r\left\lceil\frac{(k+1)m}{n}\right\rceil - m}$
OJ	$q^{(m-r)(r-1)+2}\, q^{(r-1)(k+1)}$	$q^{(m-r)(r-1)}\, q^{(r_1-1)(k-r_1)+\gamma}$ $\gamma = \max\{r_i : i \in \{2..\ell\}\}$
Annulator Polynomial	$q^{r\left\lceil\frac{(k+1)(r+1)-(n+1)}{r}\right\rceil}$ $n^{\binom{r+k+d_{reg}-1}{d_{reg}}\omega}$	$\min\left\{q^{r_\nu\left\lceil\frac{(k+1)(r_\nu+1)-(n_\nu+1)}{r_\nu}\right\rceil} : \nu \in \{1..\ell\}\right\}$ $\min\left\{n_\nu^{\binom{r_\nu+k+d_{reg}^{(\nu)}-1}{d_{reg}^{(\nu)}}\omega} : \nu \in \{1..\ell\}\right\}$
MM	$m\binom{n-p-k-1}{r}\left(\binom{n-p}{r}\right)^{\omega-1}$ $q^{ar}m\binom{n-k-1}{r}\left(\binom{n-a}{r}\right)^{\omega-1}$	$m\binom{n-p-k-1}{r}\left(\binom{n_\ell-p}{r_\ell}\prod_{i=1}^{\ell-1}\binom{n_i}{r_i}\right)^{\omega-1}$ $q^{\sum_{i=1}^{\ell} a_i r_i} m\binom{n-k-1}{r}\left(\prod_{i=1}^{\ell}\binom{n_i-a_i}{r_i}\right)^{\omega-1}$

Remark 4. The complexity analysis shows that the gain of most attacks on the ℓ-RD problem benefits from the blockwise structure of ℓ-errors. (1) the OJ and MM attacks benefits from the block-diagonal form of coefficient matrix $\boldsymbol{C}$ because the sparse $\boldsymbol{C}$ enables one to solve less variables (multivariable or linear) system; (2) the AGHT attack is limited because its cost depends on how to successfully guess a subspace that contains the support of the error; (3) the annulator polynomials attack benefits from the fact that the ℓ-errors allow to divide the ℓ-RD problem into ℓ subproblems with the smaller parameters.

For the powerful MM-$\mathbb{F}_q$ modeling, in the "underdetermined" case, an interesting result is that the complexity of solving the ℓ-RD problem allows to divide by a factor ℓ that of solving the standard RD problem.

Let $\ell|n$, $\ell|r$, $n' = n/\ell$, and $r' = r/\ell$. For both RD and ℓ-RD instances, when the parameters (m, n, k, r) satisfy respectively the "underdetermined" conditions: $m\binom{n-k-1}{r} < \binom{n}{r} - 1$ and $m\binom{n-k-1}{r} < \binom{n'}{r'}^{\ell} - 1$. The attacker chooses appropriate a and $(a_1, a_2, \ldots, a_\ell)$ such that

$$m\binom{n-k-1}{r} \geq \binom{n-a}{r} - 1 \quad \text{and} \quad m\binom{n-k-1}{r} \geq \prod_{i=1}^{\ell} \binom{n'-a_i}{r'} - 1$$

exactly hold. This means $\binom{n-a}{r} \approx \prod_{i=1}^{\ell} \binom{n'-a_i}{r'}$. From Lemma C.1 in Appendix C.6 of full version [40], an appropriate choice is $a_1 = a_2 = \cdots = a_\ell$ and $a_i = a/\ell$. At this point,

$$\frac{\log_q(T_{\text{RD}})}{\log_q(T_{\ell\text{-RD}})} \approx \frac{ar}{\sum_i^{\ell} a_i r_i} = \ell \implies T_{\ell\text{-RD}} \approx \sqrt[\ell]{T_{\text{RD}}},$$

where T_{RD} and $T_{\ell\text{-RD}}$ are the complexity of solving the RD and ℓ-RD problems, respectively. This further shows that the speedup really benefits from the block-diagonal form of $\boldsymbol{C}$ because having $\boldsymbol{C}$ sparse enables one to guess $\sum_{i=1}^{\ell} a_i r_i$ entries of $\boldsymbol{C}$ to convert the "underdetermined" system into an "overdetermined" system, instead of ar entries in the standard RD problem.

We simulate the complexity of MM-$\mathbb{F}_q$ for RD, 2-RD, and 3-RD in Fig. 1.

- (a) The RD instances are estimated with $(q, m, n, k) = (2, 200, 200, 100)$ and various even values $r = 2r'$ ($r' \in \{3..30\}$). The 2-RD instances are estimated with $(q, m, n, k, n_1, n_2) = (2, 200, 200, 100, 100, 100)$ and various values $r_1 = r_2 \in \{3..30\}$.
- (b) The RD instances are estimated with $(q, m, n, k) = (2, 100, 200, 100)$ and various even values $r \in \{6..40\}$. The 2-RD instances are estimated with $(q, m, n, k, n_1, n_2) = (2, 100, 200, 100, 100, 100)$ and various values $r_1 = r_2 \in \{3..20\}$.
- (c) The RD instances are estimated with $(q, m, n, k) = (2, 100, 300, 100)$ and various values $r = 3r'$ ($r' \in \{2..20\}$). The 3-RD instances are estimated with $(q, m, n, k, n_1, n_2, n_3) = (2, 100, 300, 100, 100, 100, 100)$ and various values $r_1 = r_2 = r_3 \in \{2..20\}$.

Our simulations become interesting as r increases. (a) and (b) in Fig. 1 show that, when r is divided equally into (r_1, r_2), the exponential complexity allows to divide by a factor 2 for $r \geq 10$, i.e., $T_{\text{2-RD}} \approx \sqrt{T_{\text{RD}}}$. (c) in Fig. 1 shows that, when r is divided equally into (r_1, r_2, r_3), the exponential complexity allows to divide by a factor 3 for $r \geq 12$, i.e., $T_{\text{3-RD}} \approx \sqrt[3]{T_{\text{RD}}}$. The parameters sizes in (b) and (c) are exactly the case of cryptography parameters in Sect. 5.

4 The ℓ-LRPC Codes and Decoding Algorithm

In this section, we define the blockwise LRPC (ℓ-LRPC) codes, give its decoding algorithm, and analyze the decoding failure probability and the error-correcting

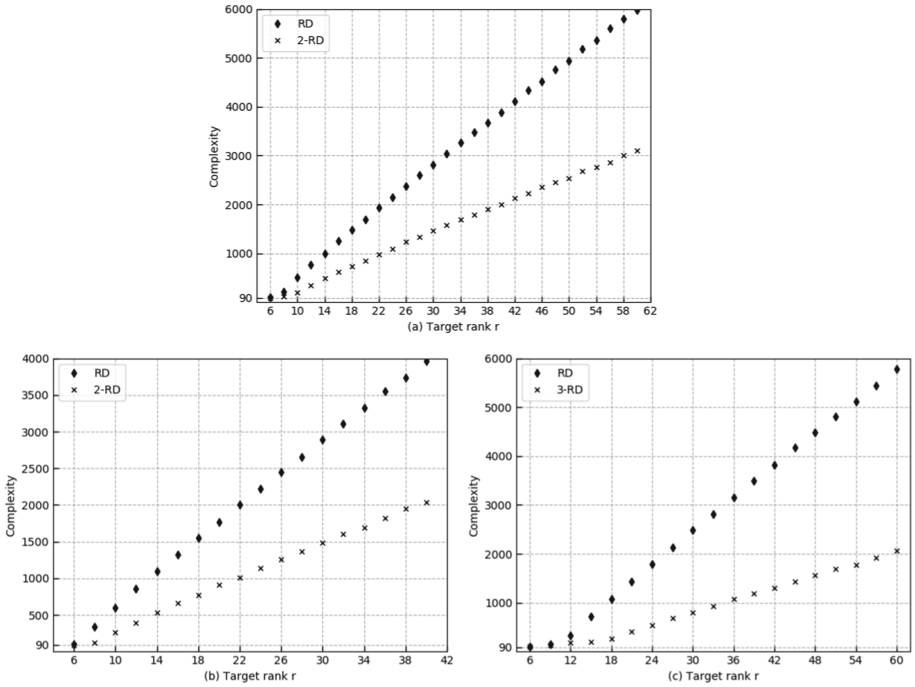

Fig. 1. Complexity trend of RD, 2-RD, and 3-RD by MM-$\mathbb{F}_q$.

capability. We find that the decoding algorithm can benefit from the blockwise structure: the decoding capacity can be significantly improved by a factor of ℓ. For cryptography applications in Sect. 5, we finally give the ℓ-Rank Support Recover (ℓ-RSR) algorithm which is used to recover the support of the ℓ-error.

4.1 The ℓ-LRPC Codes

An $[n,k]_{q^m}$ LRPC code [4,20] is defined by a parity-check matrix $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ with small weight. Our $[n,k]_{q^m}$ ℓ-LRPC code is defined by a parity-check matrix consisting of ℓ small-weight matrices of size $(n-k)\times n_i$.

Definition 4.1 (Blockwise LRPC (ℓ-LRPC) Codes). *Let $\ell, k \in \mathbb{N}$, $n_i, d_i \in \mathbb{N}$ for $i \in \{1..\ell\}$, and $n = \sum_{i=1}^{\ell} n_i$. Let $\boldsymbol{H}_i \in \mathbb{F}_{q^m}^{(n-k)\times n_i}$ be a matrix of weight d_i. Let the supports of ℓ matrices $\boldsymbol{H}_i$'s are mutually disjoint. An $[n,k]_{q^m}$ ℓ-LRPC code of length n and dimension k is defined by a parity-check matrix $\boldsymbol{H} = (\boldsymbol{H}_1\ \boldsymbol{H}_2\ \cdots\ \boldsymbol{H}_\ell) \in \mathbb{F}_{q^m}^{(n-k)\times n}$.*

Let $\boldsymbol{n} = (n_1, n_2, \ldots, n_\ell)$ and $\boldsymbol{d} = (d_1, d_2, \ldots, d_\ell)$ be vectors of positive integers. We denote the set of such parity-check matrices by $\mathcal{M}_{\boldsymbol{d}}^{\boldsymbol{n}}(k)$. Let F_i be the support of dimension d_i of $\boldsymbol{H}_i$. Because all supports are mutually disjoint,

the matrix $\boldsymbol{H}$ can be viewed as the matrix of weight $d = \sum_{i=1}^{\ell} d_i$ and support $F = \sum_{i=1}^{\ell} F_i$.

We next consider decoding algorithms for two error distributions: the ℓ-errors and the standard rank metric errors. In this subsection, we analyze the case of decoding the ℓ-errors. The decoding algorithm is also applied to ROLLO in Sect. 5. The latter is presented in Appendix D of full version [40], where we show that for the standard errors, the ℓ-LRPC code has the same decoding capacity as the standard LRPC code.

4.2 Decoding ℓ-Errors

Let $\boldsymbol{r} = (r_1, \ldots, r_\ell)$ be a vector of positive integers. Consider an $[n,k]_{q^m}$ ℓ-LRPC code $\mathcal{C}$ with generator matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k\times n}$ and parity-check matrix $\boldsymbol{H} = (\boldsymbol{H}_1\ \boldsymbol{H}_2\ \cdots\ \boldsymbol{H}_\ell) \in \mathcal{M}_{\boldsymbol{d}}^{\boldsymbol{n}}(k)$ of support $(F_1, F_2, \ldots, F_\ell)$. Let $\boldsymbol{y} = \boldsymbol{mG} + \boldsymbol{e}$ be a received word, where $\boldsymbol{m} \in \mathbb{F}_{q^m}^k$ and $\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_\ell) \in \mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ with the support $(E_1, E_2, \ldots, E_\ell)$. The syndrome $\boldsymbol{s} = \boldsymbol{Hy}^\top = \boldsymbol{He}^\top = \sum_{j=1}^{\ell} \boldsymbol{H}_j \boldsymbol{e}_j^\top$.

The general idea of decoding ℓ-error $\boldsymbol{e}$ uses the fact that the subspace $S = \langle s_1, s_2, \ldots, s_{n-k}\rangle_{\mathbb{F}_q}$ generated by $\boldsymbol{s}$ enables one to recover the space $\sum_{i=1}^{\ell} E_i F_i$. Once obtaining $\sum_{j=1}^{\ell} E_j F_j$, one recovers $E_1, E_2, \ldots, E_\ell$ and computes the support $E = \sum_{j=1}^{\ell} E_j$ of the error $\boldsymbol{e}$. Finally, the coordinates of $\boldsymbol{e}$ are computed by solving a linear system. The decoding algorithm is described in Algorithm 1.

4.3 Correctness of the Decoding Algorithm

The correctness of Algorithm 1 depends on the recovery of correct E_j, which requires $\dim S = \dim\left(\sum_{j=1}^{\ell} E_j F_j\right)$ and $\dim\left(\bigcap_{i=1}^{d_j} S_{ji}\right) = r_j$ for $j \in \{1..\ell\}$. We assume that these two conditions hold.

Step 1: the first step of the algorithm is obvious.

Step 2: we prove that $E_j = \bigcap_{i=1}^{d_j} S_{ji}$ for $j \in \{1..\ell\}$. Let $(\varepsilon_{j1}, \varepsilon_{j2}, \ldots, \varepsilon_{jr_j}) \in \mathbb{F}_{q^m}^{r_j}$ be the basis of E_j. Since $\boldsymbol{s} = \boldsymbol{He}^\top = \sum_{j=1}^{\ell} \boldsymbol{H}_j \boldsymbol{e}_j^\top$, $\boldsymbol{H} \in \mathcal{M}_{\boldsymbol{d}}^{\boldsymbol{n}}(k)$ is a matrix of support $(F_1, F_2, \ldots, F_\ell)$, and $\boldsymbol{e} \in \mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ is an ℓ-error of support $(E_1, E_2, \ldots, E_\ell)$, we have that the entries of $\boldsymbol{H}_j \boldsymbol{e}_j^\top$ respectively lie in $E_j F_j$. Thus, $S \subset \sum_{j=1}^{\ell} E_j F_j$. By assumption $\dim S = \dim\left(\sum_{j=1}^{\ell} E_j F_j\right)$, we have $S = \sum_{j=1}^{\ell} E_j F_j$. Further, for any $i \in \{1..d_j\}$, since $f_{ji}\varepsilon_{j\kappa} \in \sum_{j=1}^{\ell} E_j F_j$ for all $\kappa \in \{1..r_j\}$, we have $\varepsilon_{j\kappa} \subset S_{ji} = \{f_{ji}^{-1} x : x \in S\} \Rightarrow E_j \subset S_{ji}$. Then, $E_j \subset \bigcap_{i=1}^{d_j} S_{ji}$. By assumption $\dim\left(\bigcap_{i=1}^{d_j} S_{ji}\right) = r_j$, we have $E_j = \bigcap_{i=1}^{d_j} S_{ji}$.

Step 3: one expresses $\boldsymbol{e}$ under the basis $\boldsymbol{\varepsilon}$ of E:

$$\boldsymbol{e} = (e_1, e_2, \ldots, e_n) = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_r) \begin{pmatrix} e_{11} & e_{12} & \cdots & e_{1n} \\ e_{21} & e_{22} & \cdots & e_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ e_{r1} & e_{r2} & \cdots & e_{rn} \end{pmatrix} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_r) \begin{pmatrix} \overline{\boldsymbol{e}}_1 \\ \overline{\boldsymbol{e}}_2 \\ \vdots \\ \overline{\boldsymbol{e}}_r \end{pmatrix},$$

Algorithm 1. Decoding ℓ-errors for ℓ-LRPC codes

Input: the vector $\boldsymbol{y}$ and the parity-check matrix $\boldsymbol{H}$.
Output: the message $\boldsymbol{m}$

1: Computing syndrome space:
   – Compute the syndrome $\boldsymbol{H}\boldsymbol{y}^\top = \boldsymbol{H}\boldsymbol{e}^\top = \sum_{i=1}^{\ell} \boldsymbol{H}_i \boldsymbol{e}_i^\top = \boldsymbol{s} = (s_1, s_2, \ldots, s_{n-k})^\top$ and the syndrome space $S = \langle s_1, s_2, \ldots, s_{n-k} \rangle_{\mathbb{F}_q}$.
2: Recovering the support E of the error $\boldsymbol{e}$:
   – Compute F_j from $\boldsymbol{H}$ for $j \in \{1..\ell\}$
   – Compute the basis $(f_{j1}, f_{j2}, \ldots, f_{jd_j}) \in \mathbb{F}_{q^m}^{d_j}$ of F_j for $j \in \{1..\ell\}$
   – Compute $S_{ji} = f_{ji}^{-1} S$, where all generators of S are multiplied by f_{ji}^{-1} for $j \in \{1..\ell\}$ and $i \in \{1..d_j\}$
   – Compute $E_j = \bigcap_{i=1}^{d_j} S_{ji}$ for $j \in \{1..\ell\}$
   – Compute $E = \sum_{j=1}^{\ell} E_j$
3: Recovering the error $\boldsymbol{e}$:
   – Compute the basis $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_r) \in \mathbb{F}_{q^m}^r$ of E
   – Write each entry e_j of $\boldsymbol{e}$ as $e_j = \sum_{i=1}^{r} e_{ij} \varepsilon_j$ for $j \in \{1..n\}$ in the basis $\boldsymbol{\varepsilon}$
   – Solve e_{ij} from the linear system $\boldsymbol{H}\boldsymbol{e}^\top = \boldsymbol{s}$.
4: Recovering $\boldsymbol{m}$ from $\boldsymbol{m}\boldsymbol{G} = \boldsymbol{y} - \boldsymbol{e}$.

where $\overline{\boldsymbol{e}}_i = (e_{i1}, e_{i2}, \ldots, e_{in})$ for $i \in \{1..r\}$, and computes $\overline{\boldsymbol{e}}_i$ from Eq. (15):

$$\begin{aligned} \boldsymbol{H}\boldsymbol{e}^\top &= \begin{pmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \vdots \\ \boldsymbol{h}_{n-k} \end{pmatrix} (\overline{\boldsymbol{e}}_1^\top, \overline{\boldsymbol{e}}_2^\top, \ldots, \overline{\boldsymbol{e}}_r^\top) \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_r \end{pmatrix} \\ &= \begin{pmatrix} \boldsymbol{h}_1\varepsilon_1 & \boldsymbol{h}_1\varepsilon_2 & \cdots & \boldsymbol{h}_1\varepsilon_r \\ \boldsymbol{h}_2\varepsilon_1 & \boldsymbol{h}_2\varepsilon_2 & \cdots & \boldsymbol{h}_2\varepsilon_r \\ \vdots & \vdots & \cdots & \vdots \\ \boldsymbol{h}_{n-k}\varepsilon_1 & \boldsymbol{h}_{n-k}\varepsilon_2 & \cdots & \boldsymbol{h}_{n-k}\varepsilon_r \end{pmatrix} \begin{pmatrix} \overline{\boldsymbol{e}}_1^\top \\ \overline{\boldsymbol{e}}_2^\top \\ \vdots \\ \overline{\boldsymbol{e}}_r^\top \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_{n-k} \end{pmatrix}, \end{aligned} \tag{15}$$

where $\boldsymbol{h}_j$ is the j-th row of $\boldsymbol{H}$.

There are two methods to solve Eq. (15):

1. **Solve-$\mathbb{F}_{q^m}$:** Obtaining a linear system with nr unknowns and $m(n-k)$ equations over $\mathbb{F}_q$ by expressing $\boldsymbol{h}_j\varepsilon_i$ and s_j as a matrix $\mathrm{Mat}(\boldsymbol{h}_j\varepsilon_i) \in \mathbb{F}_q^{m \times n}$ and column vector of length m, respectively, under the basis $\boldsymbol{\alpha}$. The system has one solution with overwhelming probability if $nr \leq m(n-k)$;
2. **Solve-EF:** As $\sum_{j=1}^{\ell} E_j F_j \subset EF$, where $F = \sum_{j=1}^{\ell} F_j$, the entries of $\boldsymbol{h}_j\varepsilon_i$ and s_j lie in EF. We then can express Equation (15) under the basis of EF by expressing $\boldsymbol{h}_j\varepsilon_i$ and s_j as a matrix of $rd \times n$ and column vector of length rd, respectively. Finally, we will obtain a linear system with nr unknowns and $rd(n-k)$ equations over $\mathbb{F}_q$. The system has one solution with overwhelming probability if $nr \leq rd(n-k)$, where $d = \sum_{j=1}^{\ell} d_j$ and $r = \sum_{j=1}^{\ell} r_j$.

Once all $\overline{\boldsymbol{e}}_i$'s are obtained, one can recover $\boldsymbol{e}$. We experimentally find that **Solve-**$\mathbb{F}_{q^m}$ is more efficient than **Solve-**EF on SageMath 9.0.

Step 4: the fourth step of the algorithm is obvious.

4.4 The Decoding Complexity

The most costly part is the intersection in Step 2 and solving linear systems in Step 3. The intersection $\bigcap_{i=1}^{d_j} S_{ji}$ of spaces S_{ji} of dimension $\mu = \sum_{j=1}^{\ell} r_j d_j$ costs $\mathcal{O}\left(4\mu^2 m \sum_{j=1}^{\ell} d_j\right)$ operations in $\mathbb{F}_q$ for $j \in \{1..\ell\}$. By **Solve-**EF, expressing $\boldsymbol{h}_j \varepsilon_i$ as a matrix of $rd \times n$ in the basis of EF consists in solving n linear systems with rd unknowns and m equations. This costs $(n-k)nr^{\omega+1}d^{\omega}$ operations in $\mathbb{F}_q$. Expressing s_j as a column vector of length rd in the basis of EF consists in solving a linear system with rd unknowns and m equations. This costs $(n-k)(rd)^{\omega}$ operations in $\mathbb{F}_q$. Solving the linear system $\boldsymbol{H}\boldsymbol{e}^{\top} = \boldsymbol{s}$ with nr unknowns and $rd(n-k)$ equations costs about $\mathcal{O}((nr)^{\omega})$ operations in $\mathbb{F}_q$. Thus, the complexity of the decoding algorithm is bounded by $\mathcal{O}((nr)^{\omega})$.

4.5 Decoding Failure Probability

By the correctness assumption of Algorithm 1, two cases can make the algorithm fail: (i) $\dim S < \dim\left(\sum_{j=1}^{\ell} E_j F_j\right)$; (ii) $\dim\left(\bigcap_{i=1}^{d_j} S_{ji}\right) > r_j$ for $j \in \{1..\ell\}$. Propositions 4.2 and 4.3 estimate the probability of two cases.

Proposition 4.2. *The probability of* $\dim S < \dim\left(\sum_{j=1}^{\ell} E_j F_j\right)$ *is bounded by* $q^{-(n-k-\mu)}$ *where* $\mu = \sum_{j=1}^{\ell} r_j d_j$.

Proposition 4.3. *The probability that there is* $j \in \{1..\ell\}$ *such that* $\dim\left(\bigcap_{i=1}^{d_j} S_{ji}\right) > r_j$ *is bounded by* $\sum_{j=1}^{\ell} q^{\mu - r_j}\left(\frac{q^{\mu-r_j}-1}{q^{m-r_j}}\right)^{d_j-1}$ *where* $\mu = \sum_{j=1}^{\ell} r_j d_j$.

We give the detailed proofs for Propositions 4.2 and 4.3 in Appendices (C.8 and C.9) of full version [40]. Combining these two propositions, we deduce the decoding failure probability of Algorithm 1 in Theorem 4.4.

Theorem 4.4. *Under assumptions that* S_{ji} *behaves as independent and random subspaces containing* E_j*, the decoding failure probability of Algorithm 1 is bounded by* $q^{-(n-k-\mu)} + \sum_{j=1}^{\ell} q^{\mu - r_j}\left(\frac{q^{\mu-r_j}-1}{q^{m-r_j}}\right)^{d_j-1}$ *where* $\mu = \sum_{j=1}^{\ell} r_j d_j$.

The analysis shows that the failure probability can be made arbitrarily small.

4.6 Error Correction Capability

From the correctness of Algorithm 1, we have $nr \leq rd(n-k) \Rightarrow d \geq \frac{n}{n-k}$. Under this condition, the decoding capacity is constrained by $\sum_{j=1}^{\ell} r_j d_j \leq n-k$. The following Theorem 4.5 is obvious.

Theorem 4.5. *When $d_1 = d_2 = \cdots = d_\ell$, the ℓ-LRPC code allows to decode ℓ-errors of weight up to $r = \sum_{j=1}^{\ell} r_j = \frac{n-k}{d_1}$. By setting $d_1 = d_2 = \cdots = d_\ell = 2$, it can decode ℓ-errors of weight up to $\frac{n-k}{2}$.*

Theorem 4.5 implies that the decoding algorithm can benefit from the block-wise structure: the decoding capacity can be significantly improved by a factor of ℓ. An $[n,k]_{q^m}$ LRPC code defined by a parity-check matrix of weight d can decode the standard errors of weight up to $r = \frac{n-k}{d}$ with a DFR of about q^{rd-n-k}. Let $\ell | d$, $d_i = d/\ell$, $\boldsymbol{H} \in \mathcal{M}_{\boldsymbol{d}}^{\boldsymbol{n}}(k)$ be a parity-check matrix of an $[n,k]_{q^m}$ ℓ-LRPC code. This ℓ-LRPC code can decode ℓ-errors in $\mathcal{S}_{\boldsymbol{r}}^{\boldsymbol{n}}$ of weight up to ℓr with the same DFR, which comes from

$$\sum_{j=1}^{\ell} r_j d_j = \frac{d}{\ell} \sum_{j=1}^{\ell} r_j \leq n-k \implies \sum_{j=1}^{\ell} r_j \leq \frac{\ell(n-k)}{d} = \ell r.$$

For example, fixing $d = 4$, $r = 8$, and the DFR of q^{32-n-k}, an $[n,k]_{q^m}$ LRPC code can decode errors of weight 8, but an $[n,k]_{q^m}$ 2-LRPC codes with parameter $\boldsymbol{d} = (d_1, d_2) = (2,2)$ can decode ℓ-errors with parameter $\boldsymbol{r} = (r_1, r_2) = (8,8)$ of weight up to $r = r_1 + r_2 = 16$.

For the accurate failure probability of decoding errors of maximal weight, it is hard to estimate theoretical value and the value in Theorem 4.4 seems not practical for $q > 2$. We give a simulation of the decoding algorithm for 2-LRPC codes on SageMath 9.0. When $\ell = 2$ and $d_1 = d_2 = 2$, the 2-LRPC codes can decode 2-errors of weight up to $\frac{n-k}{2}$. The simulated result shows that the failure probability is about 0.73 for $q = 2$. Figure 2 shows the decreasing trend of the failure probability as q increases. For $q = 2$, the failure probability is bounded by $q^{-\left(n-k-\sum_{j=1}^{2} r_j d_j\right)} = 1$. For $q > 2$, the upper bound of failure probability seems to be $q^{-\left(n-k+1-\sum_{j=1}^{2} r_j d_j\right)}$. The code parameters are $(m, n, k, n_1, n_2, r_1, r_2, d_1, d_2) = (43, 44, 22, 22, 22, 6, 5, 2, 2)$ for $q = 2, 3, 5, 7, 11, 13, 17, 19$.

4.7 The ℓ-RSR Algorithm

For cryptography applications in Sect. 5, one just recovers the support of the error. In this subsection, we give the ℓ-Rank Support Recover (ℓ-RSR) algorithm (Algorithm 2), which is a shortened version of the decoding Algorithm 1 without the computation of the error. The correctness follows Algorithm 1. The failure probability follows Theorem 4.4. The cost is only the recovery of support and is given in Subsect. 4.4.

5 Applications to Cryptography

In this section, we apply the ideal variants of the ℓ-RD problem and the ℓ-LRPC codes to improve RQC [30] and ROLLO [29] kept in NIST PQC Round 2. Due to

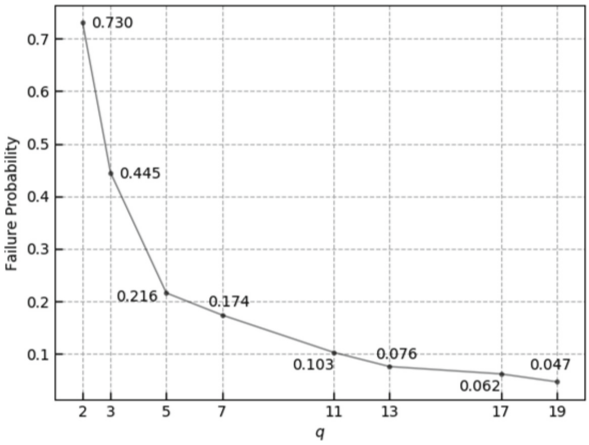

Fig. 2. Simulated failure probability of decoding 2-errors of weight $\frac{n-k}{2}$ for 2-LRPC codes.

Algorithm 2. ℓ-RSR Algorithm

Input: a parity-check matrix $\boldsymbol{H} = (\boldsymbol{H}_1\ \boldsymbol{H}_2\ \cdots\ \boldsymbol{H}_\ell) \in \mathcal{M}_{\boldsymbol{d}}^{\boldsymbol{n}}(k)$, a syndrome $\boldsymbol{s} \in \mathbb{F}_{q^m}^{n-k}$, $\boldsymbol{r} = (r_1, r_2, \ldots, r_\ell)$.
Output: ℓ spaces E_j of dimensions r_j.

1: Compute the syndrome space $S = \langle s_1, s_2, \ldots, s_{n-k} \rangle_{\mathbb{F}_q}$.
2: Recovering the support E_j for $j \in \{1..\ell\}$:
- Compute F_j from $\boldsymbol{H}_j$
- Compute the basis $(f_{j1}, f_{j2}, \ldots, f_{jd_j})$ of F_j
- Compute $S_{ji} = f_{ji}^{-1} S$, where all generators of S are multiplied by f_{ji}^{-1} for $i \in \{1..d_j\}$
- Compute $E_j = \bigcap_{i=1}^{d_j} S_{ji}$

space limitations, we present the ideal variants in Appendix E of full version [40] and only list improved schemes and comparisons in this section.

RQC [30] and ROLLO [29] include Public Key Encryptions (PKE) and Key Encapsulation Mechanisms (KEM). RQC is an IND-CCA2 KEM built from its IND-CPA PKE construction based on the HHK transformation [26] and uses the Gabidulin codes. We only consider the PKE version of RQC for simplicity. ROLLO is the merge of the three cryptosystems Laker, Locker, and Ouroboros-R which all share the same decryption algorithm for the LRPC codes. Laker (ROLLO-I) and Ouroboros-R (ROLLO-III) are two IND-CPA KEM. Locker (ROLLO-II) is an IND-CCA2 PKE scheme built from its IND-CPA PKE construction based on the HHK transformation [26]. We only consider the IND-CPA PKE version of Locker for simplicity.

5.1 Improved RQC

In this subsection, we improve RQC [30] based on the 2-IRSD and 3-IRSD problems. Our RQC uses three types of codes: a Gabidulin code $\mathcal{C}$ [18] with generator matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k\times n}$ which can correct up to $\lfloor \frac{n-k}{2} \rfloor$ errors by a deterministic decoding algorithm $\mathcal{C}.\mathsf{Decode}$ [6,27], a random $[2n, n]_{q^m}$-ideal code with parity-check matrix $(\mathbf{1} \ \ \boldsymbol{h})$, and a random $[3n, n]_{q^m}$-ideal code with parity-check matrix $\begin{pmatrix} \mathbf{1} & \mathbf{0} & \boldsymbol{h} \\ \mathbf{0} & \mathbf{1} & \boldsymbol{s} \end{pmatrix}$.

- $\mathsf{RQC.KGen}(\lambda)$: Taking 1^λ as input, it randomly samples $\boldsymbol{h} \xleftarrow{\$} \mathbb{F}_{q^m}^n$ and $(\boldsymbol{x}, \boldsymbol{y}) \xleftarrow{\$} \mathcal{S}^{(n,n)}_{(w_{\boldsymbol{x}}, w_{\boldsymbol{y}})}$, computes $\boldsymbol{s} = \boldsymbol{x} + \boldsymbol{h}\boldsymbol{y}$, and sets the public key $pk = (\boldsymbol{h}, \boldsymbol{s})$ and the private key $sk = (\boldsymbol{x}, \boldsymbol{y})$.
- $\mathsf{RQC.Enc}(pk, \boldsymbol{m})$: Taking the public key $pk = (\boldsymbol{s}, \boldsymbol{h})$ and a message $\boldsymbol{m} \in \mathbb{F}_{q^m}^k$ as input, it randomly samples $(\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{e}) \xleftarrow{\$} \mathcal{S}^{(n,n,n)}_{(w_{\boldsymbol{r}_1}, w_{\boldsymbol{r}_2}, w_{\boldsymbol{e}})}$, computes $\boldsymbol{u} = \boldsymbol{r}_1 + \boldsymbol{h}\boldsymbol{r}_2$ and $\boldsymbol{v} = \boldsymbol{m}\boldsymbol{G} + \boldsymbol{s}\boldsymbol{r}_2 + \boldsymbol{e}$, and returns the ciphertext $\boldsymbol{c} = (\boldsymbol{u}, \boldsymbol{v})$.
- $\mathsf{RQC.Dec}(sk, \boldsymbol{c})$: Taking a private key $sk = (\boldsymbol{x}, \boldsymbol{y})$ and the ciphertext $\boldsymbol{c}$ as input, it computes $\boldsymbol{v} - \boldsymbol{u}\boldsymbol{y}$ and returns $\boldsymbol{m} \leftarrow \mathcal{C}.\mathsf{Decode}(\boldsymbol{v} - \boldsymbol{u}\boldsymbol{y})$.

Fig. 3. Description of our RQC PKE scheme.

Correctness. We have $\boldsymbol{v} - \boldsymbol{u}\boldsymbol{y} = \boldsymbol{m}\boldsymbol{G} + \boldsymbol{x}\boldsymbol{r}_2 + \boldsymbol{e} - \boldsymbol{r}_1\boldsymbol{y}$. The correctness of our encryption scheme is based on the decoding capability of the Gabidulin code $\mathcal{C}$, i.e., the error term $\boldsymbol{x}\boldsymbol{r}_2 + \boldsymbol{e} - \boldsymbol{r}_1\boldsymbol{y}$ must fulfill: $\|\boldsymbol{x}\boldsymbol{r}_2 + \boldsymbol{e} - \boldsymbol{r}_1\boldsymbol{y}\|_{\mathrm{R}} = w_{\boldsymbol{x}}w_{\boldsymbol{r}_2} + w_{\boldsymbol{y}}w_{\boldsymbol{r}_1} + w_{\boldsymbol{e}} \le \lfloor \frac{n-k}{2} \rfloor$.

In the decryption step, one needs to decode an error of weight $w_{\boldsymbol{x}}w_{\boldsymbol{r}_2} + w_{\boldsymbol{y}}w_{\boldsymbol{r}_1} + w_{\boldsymbol{e}}$. This weight increase is slow, which brings the gain of decoding capacity and saves code parameters. Although the ℓ-errors can also be used to speed up the attacks for decoding problems, the performance in Table 3 shows that the gain in the decoding method greatly outweighs the gain in the attacks, and eventually allows scheme with small parameters.

Theorem 5.1. *Under the decisional 2-IRSD and 3-IRSD problems, our RQC PKE in Fig. 3 is IND-CPA secure.*

Proof. The proof is similar to [30] with 2-IRSD and 3-IRSD instances. The two instances are defined by

$$\boldsymbol{s} = (\mathbf{1} \ \ \boldsymbol{h}) \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{pmatrix}, \quad \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{v} - \boldsymbol{m}\boldsymbol{G} \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \boldsymbol{h} \\ \mathbf{0} & \mathbf{1} & \boldsymbol{s} \end{pmatrix} \begin{pmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \boldsymbol{e} \end{pmatrix}.$$

□

5.2 Improved Lake (ROLLO-I)

In this subsection, we improve Lake based on the 2-IRSD problem and the 2-ILRPC codes indistinguishability problem. Our Laker has three building blocks: a random $[2n, n]_{q^m}$ 2-ILRPC code with parity-check matrix $(\boldsymbol{x} \ \ \boldsymbol{y})$, the algorithm 2-RSR (see Algorithm 2), and a random $[2n, n]_{q^m}$-ideal code with parity-check matrix $(\mathbf{1} \ \ \boldsymbol{h})$.

- Lake.KGen(λ): Taking 1^λ as input, it samples $(\boldsymbol{x}, \boldsymbol{y}) \xleftarrow{\$} \mathcal{S}^{(n,n)}_{(d_1,d_2)}$ and computes $\boldsymbol{h} = \boldsymbol{x}^{-1}\boldsymbol{y}$, then it sets the public key $pk = \boldsymbol{h}$ and the private key $sk = (\boldsymbol{x}, \boldsymbol{y})$.
- Lake.Encap(pk): Taking the public key $\boldsymbol{h}$ as input, it randomly chooses $(\boldsymbol{e}_1, \boldsymbol{e}_2) \xleftarrow{\$} \mathcal{S}^{(n,n)}_{(r_1,r_2)}$ and computes $\boldsymbol{c} = \boldsymbol{e}_1 + \boldsymbol{h}\boldsymbol{e}_2$, $E_1 = \mathrm{Supp}(\boldsymbol{e}_1)$, $E_2 = \mathrm{Supp}(\boldsymbol{e}_2)$, $E = E_1 + E_2$, and $K = \mathrm{Hash}(E)$, and returns $(\boldsymbol{c}, K)$.
- Lake.Decap($sk, \boldsymbol{c}$): Taking $(\boldsymbol{x}, \boldsymbol{y})$ and $\boldsymbol{c}$ as input, it computes $\boldsymbol{x}\boldsymbol{c} = \boldsymbol{x}\boldsymbol{e}_1 + \boldsymbol{y}\boldsymbol{e}_2$, executes $(E_1, E_2) \leftarrow$ 2-RSR$((\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{x}\boldsymbol{c}, r_1, r_2)$, computes $E = E_1 + E_2$, and returns $K = \mathrm{Hash}(E)$.

Fig. 4. Description of our Lake KEM scheme.

5.3 Improved Locker (ROLLO-II)

Locker (ROLLO-II [29]) is a PKE scheme and is obtained from ROLLO-I. In this subsection, we improve ROLLO-II by the 2-IRSD problem. As our Lake, our Locker has three building blocks: a random $[2n, n]_{q^m}$ 2-ILRPC code with parity-check matrix $(\boldsymbol{x} \ \ \boldsymbol{y})$, the algorithm 2-RSR (see Algorithm 2), and a random $[2n, n]_{q^m}$-ideal code with parity-check matrix $(\mathbf{1} \ \ \boldsymbol{h})$.

- Locker.KGen(λ): Taking 1^λ as input, it samples $(\boldsymbol{x}, \boldsymbol{y}) \xleftarrow{\$} \mathcal{S}^{(n,n)}_{(d_1,d_2)}$ and computes $\boldsymbol{h} = \boldsymbol{x}^{-1}\boldsymbol{y}$, then it sets the public key $pk = \boldsymbol{h}$ and the private key $sk = (\boldsymbol{x}, \boldsymbol{y})$.
- Locker.Enc(pk, M): Taking the public key $\boldsymbol{h}$ and a message M as input, it randomly chooses $(\boldsymbol{e}_1, \boldsymbol{e}_2) \xleftarrow{\$} \mathcal{S}^{(n,n)}_{(r_1,r_2)}$, computes $\boldsymbol{c} = \boldsymbol{e}_1 + \boldsymbol{h}\boldsymbol{e}_2$, $E_1 = \mathrm{Supp}(\boldsymbol{e}_1)$, $E_2 = \mathrm{Supp}(\boldsymbol{e}_2)$, $E = E_1 + E_2$, and the ciphertext $C = (\boldsymbol{c}, M \oplus \mathrm{Hash}(E)) = (\boldsymbol{c}, \boldsymbol{c}')$, and returns C.
- Locker.Dec(sk, C): Taking the private key $(\boldsymbol{x}, \boldsymbol{y})$ and the ciphertext C as input, it computes $\boldsymbol{x}\boldsymbol{c} = \boldsymbol{x}\boldsymbol{e}_1 + \boldsymbol{y}\boldsymbol{e}_2$, executes $(E_1, E_2) \leftarrow$ 2-RSR$((\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{x}\boldsymbol{c}, r_1, r_2)$, computes $E = E_1 + E_2$, and returns $M = \boldsymbol{c}' \oplus \mathrm{Hash}(E)$.

Fig. 5. Description of our Locker PKE scheme.

In Laker and Locker, the decapsulation and decryption steps obtain the support of $(\boldsymbol{e}_1, \boldsymbol{e}_2)$ from $\boldsymbol{x}\boldsymbol{e}_1 - \boldsymbol{y}\boldsymbol{e}_2$ of weight $r_1d_1 + r_2d_2$. This weight increase implies

that the parameters (r_1, r_2) and (d_1, d_2) can be increased a lot. Although the 2-errors and the 2-LRPC codes can also be used to speed up the attacks for decoding problems, the performance in Tables 4, 5 and 7 shows that the gain in the decoding method outweighs the gain in the attacks, and eventually allows schemes with small parameters.

Theorem 5.2. *Under the 2-ILRPC codes indistinguishability, and 2-IRSR problems our Lake KEM in Fig. 4 and Locker PKE in Fig. 5 are IND-CPA secure in the random oracle model.*

Proof. The proofs are similar to [29] with the 2-ILRPC codes indistinguishability and 2-IRSR instances. The two instances are defined by

$$\mathbf{0} = \begin{pmatrix} \mathbf{1} & \boldsymbol{h} \end{pmatrix} \begin{pmatrix} \boldsymbol{y} \\ -\boldsymbol{x} \end{pmatrix}, \quad \boldsymbol{c} = \begin{pmatrix} \mathbf{1} & \boldsymbol{h} \end{pmatrix} \begin{pmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix}.$$

□

5.4 Improved Ouroboros-R (ROLLO-III)

In this subsection, we improve ROLLO-III based on the 2-IRSD and 3-IRSD problems. Our Ouroboros-R has three building blocks: a 3-ILRPC code with parity-check matrix $(\boldsymbol{h}_0 \ \boldsymbol{h}_1 \ \mathbf{1})$, the algorithm 3-RSR (see Algorithm 2), a $[2n, n]_{q^m}$-ideal code with parity-check matrix $(\mathbf{1} \ \boldsymbol{f}_1)$, and a $[3n, n]_{q^m}$-ideal code with parity-check matrix $\begin{pmatrix} \mathbf{1} & \mathbf{0} & \boldsymbol{f}_0 \\ \mathbf{0} & \mathbf{1} & \boldsymbol{f}_1 \end{pmatrix}$.

- Ouroboros-R.KGen(λ): Taking 1^λ as input, it samples $\boldsymbol{f}_1 \xleftarrow{\text{seed}} \mathbb{F}_{q^m}^n$, and $(\boldsymbol{h}_0, \boldsymbol{h}_1) \xleftarrow{\$} \mathcal{S}_{(d_1,d_2)}^{(n,n)}$, then it computes $\boldsymbol{f}_0 = \boldsymbol{h}_1 + \boldsymbol{f}_1\boldsymbol{h}_0$ and sets the public key $pk = (\boldsymbol{f}_0, \mathsf{seed})$ and the private key $sk = (\boldsymbol{h}_0, \boldsymbol{h}_1)$.
- Ouroboros-R.Encap(pk): Taking the public key $(\boldsymbol{f}_0, \mathsf{seed})$ as input, it randomly chooses $(\boldsymbol{e}_0, \boldsymbol{e}_1, \boldsymbol{e}) \xleftarrow{\$} \mathcal{S}_{(r_1,r_2,r_3)}^{(n,n,n)}$, computes $\boldsymbol{c}_0 = \boldsymbol{f}_0\boldsymbol{e}_1 + \boldsymbol{e}$, $\boldsymbol{c}_1 = \boldsymbol{f}_1\boldsymbol{e}_1 + \boldsymbol{e}_0$, $E_1 = \mathrm{Supp}(\boldsymbol{e}_1)$, $E_2 = \mathrm{Supp}(\boldsymbol{e}_2)$, $E = E_1 + E_2$, and $K = \mathrm{Hash}(E)$, sets $\boldsymbol{c} = (\boldsymbol{c}_0, \boldsymbol{c}_1)$, and returns $(\boldsymbol{c}, K)$.
- Ouroboros-R.Decap($sk, \boldsymbol{c}$): Taking $(\boldsymbol{h}_0, \boldsymbol{h}_1)$ and $\boldsymbol{c}$ as input, it computes $\boldsymbol{s} = \boldsymbol{c}_0 - \boldsymbol{h}_0\boldsymbol{c}_1 = -\boldsymbol{h}_0\boldsymbol{e}_0 + \boldsymbol{h}_1\boldsymbol{e}_1 + \boldsymbol{e}$, executes $(E_1, E_2) \leftarrow$ 3-RSR$((\boldsymbol{h}_0, \boldsymbol{h}_1, \mathbf{1}), \boldsymbol{s}, r_1, r_2, r_3)$, computes $E = E_1 + E_2$, and returns $K = \mathrm{Hash}(E)$.

Fig. 6. Description of our Ouroboros-R KEM scheme.

In the decapsulation step, one obtains the support of $(\boldsymbol{e}_0, \boldsymbol{e}_1)$ from $\boldsymbol{h}_1\boldsymbol{e}_1 - \boldsymbol{h}_0\boldsymbol{e}_0 + \boldsymbol{e}$ of weight $r_1d_1 + r_2d_2 + r_3$. This weight increasing implies that the parameters (r_1, r_2, r_3) and (d_1, d_2) can be increased a lot. Although the blockwise errors and LRPC codes can also be used to speed up the attacks for decoding problems, the performance in Tables 6 and 7 shows that the gain in the decoding method outweighs the gain in the attacks, and eventually allows scheme with small parameters.

Theorem 5.3. *Under the decisional* 2*-IRSD and* 3*-IRSD problems, our Ouroboros-R KEM in Fig. 6 is IND-CPA secure in the random oracle model.*

Proof. The proof is similar to [2] with the (decisional) 2-IRSD and 3-IRSD instances. The two instances are defined by

$$\boldsymbol{f}_0 = \begin{pmatrix} \mathbf{1} & \boldsymbol{f}_1 \end{pmatrix} \begin{pmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_0 \end{pmatrix}, \quad \begin{pmatrix} \boldsymbol{c}_0 \\ \boldsymbol{c}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \boldsymbol{f}_0 \\ \mathbf{0} & \mathbf{1} & \boldsymbol{f}_1 \end{pmatrix} \begin{pmatrix} \boldsymbol{e} \\ \boldsymbol{e}_0 \\ \boldsymbol{e}_1 \end{pmatrix}.$$

□

5.5 Performance and Comparison

In this subsection, we compare performance of our RQC and ROLLO with original versions.

In Tables 3, 4 and 5, parameters are chosen in two principles. First, the hardness of decoding problems (the 2-IRSD and 3-IRSD problems) is ensured to reach the target security level. The hardness is estimated by our complexity formulas. Secondly, the error-correcting capacity of rank metric codes is ensured to satisfy the decryption correctness condition. $[n,k]_{q^m}$ Gabidulin codes used in RQC require $k < n \leq m$ and correct errors of weight up to $\lfloor (n-k)/2 \rfloor$; in the decryption step, the weight of the decoded errors must $\leq \lfloor (n-k)/2 \rfloor$. The ℓ-LRPC codes used in ROLLO must satisfy a reasonable DFR in Theorem 4.4. In Tables 3 and 6, "$2n$" ("$3n$") represents the complexity of solving the 2-IRSD (3-IRSD) instances in RQC and Ouroboros-R. In Tables 4 and 5, the structural attack is estimated with parameters $(m,n,k,r_1,r_2) = \left(m, 2n - \lfloor \frac{n}{d} \rfloor, n - \lfloor \frac{n}{d} \rfloor, d_1.d_2\right)$; the message attack is estimated with parameters $(m,n,k,r_1,r_2) = (m,2n,n,r_1,r_2)$.

From Tables (3, 4, 5 and 6), our parameters sizes are smaller than those of the original ones due to the blockwise stricture, which brings a low complexity redundancy, improved the public key/ciphertext sizes, and more efficient implementations. The improved performance benefits from that the gain of using ℓ-errors and ℓ-LRPC codes in decoding capacity outweighs the complexity loss in solving the ℓ-RD problem. As an example, we provide concrete timings of implementations for our ROLLO and original versions (Table 7). The benchmark is performed on Intel(R) Core(TM) i5-7440HQ CPU@ 3.40 GHz with SageMath 9.0. The tests are available online at https://github.com/YCSong232431/NH-ROLLO. Note that, we do not compare with most recent works [12,32], where the authors constructed a series of efficient PKE and KEM schemes without ideal structure by proposing augmented Gabidulin codes and LRPC codes with multiple syndromes. Our techniques are different from [12,32] and we only consider cryptosystems with ideal structure and one syndrome.

Table 3. Comparison of parameters and sizes for RQC.

Schemes	m	n	k	w_x	w_y	w_{r_1}	w_{r_2}	w_e	pks (bytes)	cts (bytes)	total (KB)	Attack $(2n, 3n)$	Security
Our RQC	83	79	7	4	4	4	4	4	860	1704	2.5	$(2^{130}, 2^{163})$	128
Our RQC	127	113	3	5	5	5	5	5	1834	3652	5.3	$(2^{258}, 2^{214})$	192
Our RQC	139	137	5	5	5	6	6	6	2421	4826	7.1	$(2^{271}, 2^{274})$	256

Schemes	m	n	k	w_x	w_y	w_{r_1}	w_{r_2}	w_e	pks (bytes)	cts (bytes)	total (KB)	Security
RQC (NIST [30])	127	113	3	7	7	7	7	13	1834	3652	5.3	128
RQC (NIST [30])	151	149	5	8	8	8	8	16	2853	5690	8.3	192
RQC (NIST [30])	181	179	3	9	9	9	9	16	4090	8164	12.0	256

pks: $\left(\lceil \frac{mn}{8} \rceil + 40\right)$ bytes; cts: $\left(2\lceil \frac{mn}{8} \rceil + 64\right)$ bytes; total = pks + cts.

Table 4. Comparison of parameters and sizes for Lake (ROLLO-I).

Schemes	m	n	r_1	r_2	d_1	d_2	DFR	pks/cts (bytes)	Structural attack $\boldsymbol{y} - \boldsymbol{xh} = \boldsymbol{0}$	Message attack $\boldsymbol{c} = \boldsymbol{e}_1 + \boldsymbol{he}_2$	Security
Our Lake	61	67	4	4	5	4	2^{-31}	511	2^{160}	2^{144}	128
Our Lake	71	79	5	5	5	5	2^{-29}	702	2^{225}	2^{255}	192
Our Lake	79	89	5	5	6	5	2^{-34}	879	2^{281}	2^{266}	256

Schemes	m	n	r	d	DFR	pks/cts (bytes)	Security
Lake (NIST [29])	67	83	7	8	2^{-28}	696	128
Lake (NIST [29])	79	97	8	8	2^{-34}	958	192
Lake (NIST [29])	97	113	9	9	2^{-33}	1371	256

pks: $\lceil \frac{mn}{8} \rceil$ bytes. cts: $\lceil \frac{mn}{8} \rceil$ bytes.

Table 5. Comparison of parameters and sizes for Locker (ROLLO-II).

Schemes	m	n	r_1	r_2	d_1	d_2	DFR	pks (bytes)	cts (bytes)	Structural attack $\boldsymbol{y} - \boldsymbol{xh} = \boldsymbol{0}$	Message attack $\boldsymbol{c} = \boldsymbol{e}_1 + \boldsymbol{he}_2$	Security
Our Locker	89	163	4	4	4	4	2^{-131}	1814	1942	2^{134}	2^{139}	128
Our Locker	97	179	4	5	5	5	2^{-134}	2171	2299	2^{254}	2^{231}	192
Our Locker	101	181	5	5	5	5	2^{-131}	2286	2414	2^{267}	2^{357}	256

Schemes	m	n	r	d	DFR	pks (bytes)	cts (bytes)	Security
Locker (NIST [29])	83	189	7	8	2^{-134}	1941	2089	128
Locker (NIST [29])	97	193	8	8	2^{-130}	2341	2469	192
Locker (NIST [29])	97	211	8	9	2^{-136}	2559	2687	256

pks: $\lceil \frac{mn}{8} \rceil$ bytes; cts: $\lceil \frac{mn}{8} \rceil + 64$ bytes. To obtain the IND-CCA2 security, another hash is added to the ciphertext such that cts = $\lceil \frac{mn}{8} \rceil + 2 * 64$ bytes.

Table 6. Comparison of parameters and sizes for Ouroboros-R (ROLLO-III).

Schemes	m	n	r_1	r_2	r_3	d_1	d_2	DFR	pks (bytes)	cts (bytes)	Attacks $(2n, 3n)$	Security
Our Ouroboros-R	53	79	4	4	5	4	4	2^{-33}	623	1166	$(2^{147}, 2^{175})$	128
Our Ouroboros-R	89	101	6	6	6	4	5	2^{-33}	1164	2248	$(2^{196}, 2^{266})$	192
Our Ouroboros-R	97	109	6	6	7	5	5	2^{-42}	1362	2644	$(2^{275}, 2^{308})$	256

Schemes	m	n	w	w_r	δ	DFR	pks (bytes)	cts (bytes)	Security
Ouroboros-R (TIT [2])	67	83	7	7	7	2^{-28}	736	1431	128
Ouroboros-R (TIT [2])	107	113	9	9	9	2^{-24}	1552	3023	192
Ouroboros-R (TIT [2])	149	151	11	11	11	2^{-20}	2853	5625	256

pks: $\left(\lceil \frac{mn}{8} \rceil + 40\right)$ bytes and cts: $\lceil \frac{2mn}{8} \rceil$ bytes. We update DFR of Ouroboros-R.

Table 7. Timings comparisons of our ROLLO and original ROLLO.

Schemes	KGen (ms)	Encap (ms)	Decap (ms)	Security
Our Lake	715	73	257	128
Our Lake	737	100	499	192
Our Lake	1020	118	553	256
Lake (NIST [29])	995	109	391	128
Lake (NIST [29])	1220	134	525	192
Lake (NIST [29])	1390	181	838	256

Schemes	KGen (ms)	Enc (ms)	Dec (ms)	Security
Our Locker	2300	232	388	128
Our Locker	2940	280	614	192
Our Locker	3210	301	644	256
Locker (NIST [29])	2760	258	446	128
Locker (NIST [29])	3410	314	583	192
Locker (NIST [29])	2780	333	715	256

Schemes	KGen (ms)	Encap (ms)	Decap (ms)	Security
Our Ouroboros-R	101	120	246	128
Our Ouroboros-R	206	247	633	192
Our Ouroboros-R	224	262	798	256
Ouroboros-R (TIT [2])	130	153	368	128
Ouroboros-R (TIT [2])	275	308	1040	192
Ouroboros-R (TIT [2])	504	614	2560	256

6 Conclusion and Future Work

In this paper, we studied blockwise structures in rank-based cryptosystems and introduced ℓ-errors, ℓ-RD problem, and ℓ-LRPC codes. They are natural generalizations of the standard errors, RD problem, and LRPC codes. We found that (1) the blockwise structure does not ease the problem too much: the ℓ-RD prob-

lem is still exponentially hard for appropriate choices of $\ell > 1$; (2) the decoding algorithm can benefit from the blockwise structure: the decoding capacity can be significantly improved by a factor of ℓ. Interestingly, the gain of the decoding capacity outweighs the complexity loss in solving the ℓ-RD problem, which allows to improve RQC and ROLLO. For 128-bit security, our RQC has total public key and ciphertext sizes of 2.5 KB, which is not only about 50% more compact than the original RQC, but also smaller than the NIST Round 4 code-based submissions HQC, BIKE, and Classic McEliece.

Recent works [3,12,32] proposed unstructured PKE and KEM without ideal structure for more reliable security. We would in next work analyze the complexity of blockwise rank support learning problem and apply the ℓ-LRPC codes with multiple syndromes to improve unstructured schemes.

Acknowledgement. We would like to thank the anonymous reviewers of ASIACRYPT 2023 for their helpful comments and suggestions on earlier versions of our paper. Jiang Zhang, the corresponding author, is supported by the National Key Research and Development Program of China (Grant No. 2022YFB2702000), and by the National Natural Science Foundation of China (Grant Nos. 62022018, 61932019). Xinyi Huang is supported by the National Natural Science Foundation of China (Grant No. 62032005). Wei Wu is supported by the National Natural Science Foundation of China (Grant No. 62372108). This research is also funded in part by the National Natural Science Foundation of China (Grant No. 62172096).

References

1. Alekhnovich, M.: More on average case vs approximation complexity. In: Proceedings of the 44th Symposium on Foundations of Computer Science (FOCS), pp. 298–307. IEEE Computer Society (2003)
2. Aragon, N., Blazy, O., Deneuville, J., Gaborit, P., Zémor, G.: Ouroboros: an efficient and provably secure KEM family. IEEE Trans. Inf. Theory **68**(9), 6233–6244 (2022)
3. Aragon, N., Dyseryn, V., Gaborit, P., Loidreau, P., Renner, J., Wachter-Zeh, A.: LowMS: a new rank metric code-based KEM without ideal structure. IACR Cryptology ePrint Archive, p. 1596 (2022). https://eprint.iacr.org/2022/1596
4. Aragon, N., Gaborit, P., Hauteville, A., Ruatta, O., Zémor, G.: Low rank parity check codes: new decoding algorithms and applications to cryptography. IEEE Trans. Inf. Theory **65**(12), 7697–7717 (2019)
5. Aragon, N., Gaborit, P., Hauteville, A., Tillich, J.: A new algorithm for solving the rank syndrome decoding problem. In: International Symposium on Information Theory (ISIT), pp. 2421–2425. IEEE (2018)
6. Augot, D., Loidreau, P., Robert, G.: Generalized Gabidulin codes over fields of any characteristic. Des. Codes Crypt. **86**(8), 1807–1848 (2018)
7. Bardet, M., et al.: An algebraic attack on rank metric code-based cryptosystems. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 64–93. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_3
8. Bardet, M., Briaud, P., Bros, M., Gaborit, P., Tillich, J.: Revisiting algebraic attacks on MinRank and on the rank decoding problem. IACR Cryptology ePrint Archive, p. 1031 (2022). https://eprint.iacr.org/2022/1031

9. Bardet, M., et al.: Improvements of algebraic attacks for solving the rank decoding and MinRank problems. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020. LNCS, vol. 12491, pp. 507–536. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64837-4_17
10. Bernstein, D.J., Chou, T., Cid, C., et al.: Classic McEliece. Fourth Round Submission to the NIST Post-quantum Cryptography Call (2022). https://classic.mceliece.org/
11. Bettaieb, S., Bidoux, L., Connan, Y., Gaborit, P., Hauteville, A.: The Learning with Rank Errors problem and an application to symmetric authentication. In: International Symposium on Information Theory, ISIT, pp. 2629–2633. IEEE (2018)
12. Bidoux, L., Briaud, P., Bros, M., Gaborit, P.: RQC revisited and more cryptanalysis for rank-based cryptography. CoRR (2022). https://doi.org/10.48550/arXiv.2207.01410
13. Byrne, E., Gluesing-Luerssen, H., Ravagnani, A.: Fundamental properties of sum-rank-metric codes. IEEE Trans. Inf. Theory **67**(10), 6456–6475 (2021)
14. Chabaud, F., Stern, J.: The cryptographic security of the syndrome decoding problem for rank distance codes. In: Kim, K., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, pp. 368–381. Springer, Heidelberg (1996). https://doi.org/10.1007/BFb0034862
15. Coggia, D., Couvreur, A.: On the security of a Loidreau rank metric code based encryption scheme. Des. Codes Crypt. **88**(9), 1941–1957 (2020)
16. Couvreur, A., Gaborit, P., Gauthier-Umaña, V., Otmani, A., Tillich, J.: Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes. Des. Codes Crypt. **73**(2), 641–666 (2014)
17. Faugère, J.-C., Levy-dit-Vehel, F., Perret, L.: Cryptanalysis of MinRank. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 280–296. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85174-5_16
18. Gabidulin, E.M.: Theory of codes with maximum rank distance. Problemy Peredachi Informatsii **21**(1), 3–16 (1985)
19. Gabidulin, E.M., Paramonov, A.V., Tretjakov, O.V.: Ideals over a non-commutative ring and their application in cryptology. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 482–489. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-46416-6_41
20. Gaborit, P., Murat, G., Ruatta, O., Zémor, G.: Low rank parity check codes and their application to cryptography. In: The Workshop on Coding and Cryptography (WCC) (2013). http://www.selmer.uib.no/WCC2013/pdfs/Gaborit.pdf
21. Gaborit, P., Ruatta, O., Schrek, J.: On the complexity of the rank syndrome decoding problem. IEEE Trans. Inf. Theory **62**(2), 1006–1019 (2016)
22. Gaborit, P., Ruatta, O., Schrek, J., Zémor, G.: New results for rank-based cryptography. In: Pointcheval, D., Vergnaud, D. (eds.) AFRICACRYPT 2014. LNCS, vol. 8469, pp. 1–12. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06734-6_1
23. Gaborit, P., Zémor, G.: On the hardness of the decoding and the minimum distance problems for rank codes. IEEE Trans. Inf. Theory **62**(12), 7245–7252 (2016)
24. Ghatak, A.: Extending Coggia-Couvreur attack on Loidreau's rank-metric cryptosystem. Des. Codes Crypt. **90**(1), 215–238 (2022)
25. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: a ring-based public key cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998 (ANTS-III). LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054868

26. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10677, pp. 341–371. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_12
27. Loidreau, P.: A Welch–Berlekamp like algorithm for decoding Gabidulin codes. In: Ytrehus, Ø. (ed.) WCC 2005. LNCS, vol. 3969, pp. 36–45. Springer, Heidelberg (2006). https://doi.org/10.1007/11779360_4
28. Loidreau, P.: A new rank metric codes based encryption scheme. In: Lange, T., Takagi, T. (eds.) PQCrypto 2017. LNCS, vol. 10346, pp. 3–17. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59879-6_1
29. Melchor, C.A., Aragon, N., Bardet, M., et al.: ROLLO. Second Round Submission to the NIST Post-quantum Cryptography Call (2020). https://pqc-rollo.org/
30. Melchor, C.A., Aragon, N., Bettaieb, S., et al.: RQC. Second Round Submission to the NIST Post-quantum Cryptography Call (2020). http://pqc-rqc.org/
31. Melchor, C.A., Aragon, N., Bettaieb, S., et al.: HQC. Fourth Round Submission to the NIST Post-quantum Cryptography Call (2023). http://pqc-hqc.org
32. Melchor, C.A., Aragon, N., Dyseryn, V., Gaborit, P., Zémor, G.: LRPC codes with multiple syndromes: near ideal-size KEMs without ideals. In: Cheon, J.H., Johansson, T. (eds.) Post-Quantum Cryptography (PQCrypto), vol. 13512, pp. 45–68. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-17234-2_3
33. Misoczki, R., Tillich, J., Sendrier, N., Barreto, P.S.L.M.: MDPC-McEliece: new McEliece variants from moderate density parity-check codes. In: IEEE International Symposium on Information Theory (ISIT), pp. 2069–2073. IEEE (2013)
34. NIST: Status report on the second round of the NIST post-quantum cryptography standardization process (2020). https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8309.pdf
35. NIST: Status report on the third round of the NIST post-quantum cryptography standardization process (2022). https://doi.org/10.6028/NIST.IR.8413-upd1
36. Ore, O.: On a special class of polynomials. Trans. Am. Math. Soc. **35**(3), 559–584 (1933)
37. Ourivski, A.V., Johansson, T.: New technique for decoding codes in the rank metric and its cryptography applications. Probl. Inf. Transm. **38**(3), 237–246 (2002)
38. Overbeck, R.: A new structural attack for GPT and variants. In: Dawson, E., Vaudenay, S. (eds.) Mycrypt 2005. LNCS, vol. 3715, pp. 50–63. Springer, Heidelberg (2005). https://doi.org/10.1007/11554868_5
39. Sidelnikov, V.M., Shestakov, S.O.: On insecurity of cryptosystems based on generalized Reed-Solomon codes. Discret. Appl. Math. **2**(4), 439–444 (1992)
40. Song, Y., Zhang, J., Huang, X., Wu, W.: Blockwise rank decoding problem and LRPC codes: cryptosystems with smaller sizes. Cryptology ePrint Archive, Paper 2023/1387 (2023). https://eprint.iacr.org/2023/1387