Rank Syndrome Decoding Estimator – An Asymptotic and Concrete Analysis

Andre Esser[1], Javier Verbel[1], and Ricardo Villanueva-Polanco[2]

[1] Technology Innovation Institute, UAE
{andre.esser,javier.verbel}@tii.ae
[2] Universidad del Norte, Colombia
rpolanco@uninorte.edu.co

Abstract. The Rank Syndrome Decoding (RSD) problem forms the foundation of many post-quantum cryptographic schemes. Its inherent hardness, with best known algorithms for common parameter regimes running in time exponential in n^2 (for n being the code length), enables compact parameter choices and efficient constructions. Several RSD-based submissions to the first NIST PQC process in 2017 were, however, invalidated by algebraic attacks, raising fundamental concerns about the security of RSD-based designs.
In this work, we revisit the parameters of prominent rank-based constructions and analyze the rationales that guided their selection, as well as their security against modern attacks. We provide a unified complexity analysis of all major RSD algorithms, including combinatorial, algebraic, and hybrid approaches, under a common cost model. All estimates are made publicly available through a dedicated open source module.
Furthermore, we present the first asymptotic analysis of these algorithms, yielding deep insights into the relations between different procedures. We show that all studied algorithms converge to one of three distinct asymptotic runtime exponents. We then provide an asymptotic baseline in terms of the worst-case decoding exponent. In particular, we find that for an extension degree equal to the code length, the best known algorithms achieve a complexity of $2^{0.1481n^2+o(n^2)}$, attained simultaneously by algebraic and combinatorial approaches. Overall, our results reinforce confidence in the RSD assumption and the design rationales of modern RSD-based schemes such as RYDE.

1 Introduction

The Rank Syndrome Decoding (RSD) problem has long served as the foundation for a variety of code-based cryptosystems, though its prominence has fluctuated over time. It is considered a promising post-quantum assumption, with several schemes submitted to the first NIST PQC standardization process in 2017 [31,32] and, more recently, to the renewed call for digital signatures through the RYDE signature scheme [5].

Informally, the Rank Syndrome Decoding (RSD) problem asks to recover a vector $\mathbf{e} \in \mathbb{F}_{q^m}^n$ of small rank weight r satisfying $\mathbf{He}^\top = \mathbf{s}$, where $\mathbf{H}$ is the

parity-check matrix of a $\mathbb{F}_{q^m}$-linear code of length n and dimension k. Since the best known algorithms for solving RSD have exponential complexity in n^2, the problem is a natural candidate for post-quantum cryptography: This high complexity allows for smaller parameter sizes and leads to compact signatures, ciphertexts, and keys compared to constructions in the Hamming metric.

Despite these advantages, the history of rank-based schemes is marked by a major setback. Algebraic attacks [7,9] during the first NIST PQC process revealed substantial weaknesses in several submissions and caused their elimination in round two of the process [2]. This event caused a lasting image loss for the RSD problem, and a certain degree of skepticism persists within the cryptographic community since then. As new proposals emerge, it is important to understand whether modern schemes are built upon more resilient design principles and parameter rationales, or whether similar vulnerabilities could evolve.

In the meantime, substantial progress has been made in the analysis of algorithms solving the RSD problem [8,20]. These works have refined complexity estimates and improved our understanding of the underlying structure. However, the security evaluation of concrete schemes remains largely ad-hoc, lacking a unified framework for evaluation. In other post-quantum settings, such as the Hamming-metric Syndrome Decoding (SD) [22], Multivariate Quadratic (MQ) [14], or Learning with Errors (LWE) [3] problems, the availability of community-maintained estimators has proven highly beneficial. They provide transparent, comparable, and reproducible security estimates. Establishing a similar, centralized estimator for the RSD problem would thus be an important step towards more consistent evaluations and stronger confidence in modern RSD-based designs.

So far, most existing analyses have focused on bit-level complexity estimates. While these are essential for setting concrete parameters, understanding the asymptotic behavior of RSD algorithms is equally important. Asymptotic analysis helps in revealing relationships between different algorithms, highlight promising directions for improvement, and show where progress may still be possible. In the Hamming metric, for example, the study of asymptotic exponents has driven much of the algorithmic progress of the past decades [12,18,21,23,29,30]. Similar trends can be observed in the lattice setting with respect to improved sieving algorithms [10,11,33,37].

Motivated by this, we aim to provide both, unified complexity estimates and a systematic asymptotic analysis for all major RSD algorithms.

Our Contributions In this work, we present a comprehensive study of the complexity of RSD algorithms, bridging practical bit-level estimates and asymptotic behavior.

Comprehensive survey and estimates. We first give a detailed overview of the most efficient known algorithms, covering combinatorial, algebraic, and hybrid approaches. Their bit complexities are analyzed with respect to a unified cost model, counting $\mathbb{F}_q$-multiplications as the basic operation. All estimates are

implemented in an open-source module allowing for transparent and reproducible security estimates.

Revisiting parameters. We revisit parameter sets proposed in the literature, ranging from the disqualified submissions of the 2017 NIST PQC process to modern designs such as RYDE. The analysis highlights the impact of different algorithms on these parameters, particularly the influence of algebraic attacks on the first-round NIST candidates. We confirm that the complexity of algebraic algorithms in the regime $r = \Theta(\sqrt{n})$ relevant for the 2017 NIST submissions still scales as $q^{\mathcal{O}(n^{3/2})}$, i.e., subexponential in n^2. However, the choice of particularly small constants in those constructions led to a practical rank weight of $r < \log n$. We show that the running time of algebraic approaches in this regime becomes quasi-polynomial, which led to the significant security reductions. On the other hand we re-confirm that the complexity of all known algorithms in the regime $m, k, r = \Theta(n)$ linear in n, which marks the regime of modern RSD schemes, remains fully exponential in n^2.

Asymptotic analysis. In addition to concrete estimates, we perform the first full asymptotic analysis of all major RSD algorithms. For the regime $m, k, r = \Theta(n)$, we derive their asymptotic runtime exponents c in

$$T = 2^{cn^2+o(n)},$$

where c is a function of q, m, k and r. This reveals fundamental relations between algorithms: all nine studied approaches converge to one of three asymptotic exponents, and for the common case $m = n$, only two distinct runtime classes remain.

We further establish a baseline in the form of the worst-case decoding exponent. For $m = n$, $q = 2$, and a rank weight r meeting the Gilbert–Varshamov bound, we find that the smallest known decoding exponent among all algorithms satisfies

$$\tilde{c} = \max_k c(q, m, k, r) = 0.1481,$$

achieved simultaneously by algebraic algorithms [8, 9], GRS-style algorithms [4, 20, 27], and a combinatorial algorithm by Ourivski–Johansson [35].

Overall, our results confirm that the parameters of recent RSD-based designs, including the RYDE NIST submission, are well justified. The bit-complexity estimates developed in this work have already guided their parameter selection and further reinforce the soundness of RSD as a post-quantum assumption.

Artifacts. All code associated with the RSD estimator presented in this work is available in the RankSDEstimator directory of the CryptographicEstimators library (version 2.1.1), accessible at https://github.com/Crypto-TII/CryptographicEstimators.

2 Preliminaries

Let $\mathbb{F}_q$ denote the finite field of q elements and $\mathbb{F}_{q^m}$ the field extension of $\mathbb{F}_q$ of degree m. We use $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_m)$ to denote an $\mathbb{F}_q$-basis of $\mathbb{F}_{q^m}$, where $\varphi_l : \mathbb{F}_{q^m} \to \mathbb{F}_q$ denotes the l-th projection map, i.e. $\varphi_l\left(\sum_{j=1}^{m} x_j\beta_j\right) = x_l$.

We denote by $\mathbf{I}_k$ the identity matrix of dimension k. We call $w_R(\mathbf{e})$ the *rank weight* of $\mathbf{e} = (e_1, \ldots, e_n) \in \mathbb{F}_{q^m}^n$, and it is defined as the rank of the matrix $\mathbf{X} = (e_{i,j})_{i,j} \in \mathbb{F}_q^{m\times n}$ and $e_j = e_{1,j}\beta_1 + \cdots + e_{m,j}\beta_m$. Furthermore, we let $\mathsf{Supp}(\mathbf{e}) = \langle e_1, \ldots, e_n\rangle$ be the support of $\mathbf{e}$, i.e. the $\mathbb{F}_q$-vector space spanned by its coordinates e_j. We use $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_r)$ to denote a basis of $\mathsf{Supp}(\mathbf{e})$, where $\mathbf{e} \in \mathbb{F}_{q^m}^n$ with $w_R(\mathbf{e}) = r$. We let $\mathsf{Gr}(r, \mathbb{F}_{q^m})$ denote the set of r-dimensional vectors spaces in $\mathbb{F}_{q^m}$ and correspondingly the Gaussian coefficient gives the size of $\mathsf{Gr}(r, \mathbb{F}_{q^m})$ as

$$\begin{bmatrix} m \\ r \end{bmatrix}_q = |\mathsf{Gr}(r, \mathbb{F}_{q^m})| = \prod_{i=0}^{r-1} \frac{q^m - q^i}{q^r - q^i}.$$

A *linear* (n, k)*-code* $\mathcal{C}$ over $\mathbb{F}_{q^m}$ is a k-dimensional vector subspace of $\mathbb{F}_{q^m}^n$. Such a code can be compactly represented via a generator or parity-check matrix. A matrix $\mathbf{G} \in \mathbb{F}_{q^m}^{k\times n}$ is a *generator matrix* of a linear code $\mathcal{C}$ over $\mathbb{F}_{q^m}$ if its rows form a basis of $\mathcal{C}$. A matrix $\mathbf{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ is a *parity-check* matrix of a linear $\mathcal{C}$ over $\mathbb{F}_{q^m}$ if it is full rank and $\mathcal{C} = \{\mathbf{e} \in \mathbb{F}_{q^m}^n : \mathbf{H} \cdot \mathbf{e}^\top = 0\}$. We use standard Landau notation for our complexity statements with $\tilde{\mathcal{O}}$-notation suppressing poly-logarithmic factors.

Throughout this work, we analyze the complexity of the Rank Decoding (RD) problem and its variants.

*Problem 2.1 (Rank Decoding (*RD*) Problem).* Let q, m, n, k, r be positive integers with $k < n$ and $r < m$. Given a generator matrix $\mathbf{G} \in \mathbb{F}_{q^m}^{k\times n}$ and a received word $\mathbf{y} \in \mathbb{F}_{q^m}^n$, the goal is to find a vector $\mathbf{e} \in \mathbb{F}_{q^m}^n$ such that

$$w_R(\mathbf{e}) \leq r \quad \text{and} \quad \mathbf{y} - \mathbf{e} \in \mathcal{C},$$

where $\mathcal{C} \subseteq \mathbb{F}_{q^m}^n$ denotes the linear code generated by $\mathbf{G}$. We refer to this problem as $\mathrm{RD}(q, m, n, k, r)$.

Motivated by cryptographic constructions, we assume that there is always at least one solution. Since the hardness of the problem usually grows in r, we furthermore assume $w_R(\mathbf{e}) = r$.

A commonly studied variant of the RD problem is its dual formulation — the Rank Syndrome Decoding (RSD) problem.

*Problem 2.2 (Rank Syndrome Decoding (*RSD*) Problem).* Let q, m, n, k, r be positive integers with $k < n$ and $r < m$. Given a parity-check matrix $\mathbf{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ and a syndrome vector $\mathbf{s} \in \mathbb{F}_{q^m}^{n-k}$, the goal is to find a vector $\mathbf{e} \in \mathbb{F}_{q^m}^n$ such that:

$$w_R(\mathbf{e}) \leq r \quad \text{and} \quad \mathbf{H}\mathbf{e}^\top = \mathbf{s}^\top.$$

We refer to this problem as $\mathrm{RSD}(q, m, n, k, r)$.

It is well known that the RD problem and the RSD problem are polynomial-time equivalent. Given an instance $(\mathbf{H}, \mathbf{s})$ of $\mathrm{RSD}(q, m, n, k, r)$, one can efficiently construct the corresponding instance $(\mathbf{G}, \mathbf{y})$ of $\mathrm{RD}(q, m, n, k, r)$, and vice versa. Consequently, an instance of RSD can be solved efficiently if and only if the corresponding RD instance can be solved efficiently.

A structured variant of RSD A structured variant of the RSD problem, known as RSD_s has found applications in the construction of cryptographic primitives, such as the RYDE signature scheme. While the problem has been proven to be computationally equivalent to the standard RSD problem [15] the resulting computational gap alters security estimates and requires a separate treatment.

Problem 2.3 (RSD_s *Problem).* Let q, m, n, k, r be positive integers with $k < n$ and $r < m$. Given a parity-check matrix $\mathbf{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ and a syndrome vector $\mathbf{s} \in \mathbb{F}_{q^m}^{n-k}$, the goal is to find a vector $\mathbf{e} \in \mathbb{F}_{q^m}^{n}$ such that:

$$w_R(\mathbf{e}) \leq r, \quad e_1 = 1, \quad \mathsf{Supp}(\mathbf{e}) = \langle e_1, \ldots, e_r \rangle, \quad \text{and} \quad \mathbf{H}\mathbf{e}^\top = \mathbf{s}^\top.$$

We refer to this problem as $\mathrm{RSD}_s(q, m, n, k, r)$.

Note that the essential difference to RSD lies in the fact that the first coordinate of $\mathbf{e}$ is known to be 1 and the support of the error vector is guaranteed to be generated by its first r coordinates. The following proposition connects an instance $\mathrm{RSD}_s(q, m, n, k+1, r)$ with an instance $\mathrm{RSD}(q, m, n, k, r)$.

Proposition 2.1 (Connection between RSD and RSD_s**, Proposition 1 of [15]).** *Let q, m, n, k, r be positive integers with $n > k$, and let $\mathcal{A}_s$ be an algorithm that solves an $\mathrm{RSD}_s(q, m, n, k+1, r)$ instance in time t with success probability ϵ_s. Then, there exists an algorithm $\mathcal{A}$ that solves an $\mathrm{RSD}(q, m, n, k, r)$ instance in time t with success probability at least*

$$\epsilon \geq \left(\prod_{i=0}^{r-1} \frac{q^n - q^{n-r+i}}{q^n - q^i} \right) \epsilon_s,$$

under the assumption that the linear code $\mathcal{C}$ defined by the parity-check matrix $\mathbf{H}$ of the RSD *instance contains no codewords of rank weight r.*

Alternatively, Proposition 2.1 can be rewritten to obtain a lowerbound for the expected time complexity T_{RSD_s} of solving $\mathrm{RSD}_s(q, m, n, k+1, r)$ in dependence on the expected time complexity T_{RSD} of solving $\mathrm{RSD}(q, m, n, k, r)$ as

$$T_{\mathrm{RSD}_s} \geq \left(\prod_{i=0}^{r-1} \frac{q^n - q^{n-r+i}}{q^n - q^i} \right) T_{\mathrm{RSD}}.$$

The Augmented Instance Several algorithms targeting the RSD problem operate (implicitly) on instances with zero syndrome, i.e., they are searching for a codeword of rank weight r in the underlying code, by exploiting a known reduction.

Let $(\mathbf{H}, \mathbf{s})$ be an $\mathsf{RSD}(q, m, n, k, r)$ instance such that the linear code defined by the parity-check matrix $\mathbf{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ contains no codewords of rank weight at most r; that is, there is no $\mathbf{e} \in \mathbb{F}_{q^m}^n$ satisfying $w_R(\mathbf{e}) \leq r$ and $\mathbf{H}\mathbf{e}^\top = \mathbf{0}$. Then we can transform the instance $(\mathbf{H}, \mathbf{s})$ into an equivalent instance $(\tilde{\mathbf{H}}, \mathbf{0})$ of $\mathsf{RSD}(q, m, n, k+1, r)$.

Reduction to the augmented instance. We call $\tilde{\mathcal{C}}$ an *augmented code* of the code $\mathcal{C}$ defined by the parity-check matrix $\mathbf{H}$ and corresponding generator matrix $\mathbf{G}$ if it is generated by a matrix of the form

$$\tilde{\mathbf{G}} = \begin{bmatrix} \mathbf{G} \\ \mathbf{y} \end{bmatrix} \in \mathbb{F}_{q^m}^{(k+1)\times n},$$

where $\mathbf{y} \in \mathbb{F}_{q^m}^n$ satisfies $\mathbf{H}\mathbf{y}^\top = \mathbf{s}^\top$. Let $\tilde{\mathbf{H}}$ be a parity-check matrix of $\tilde{\mathcal{C}}$. We refer to $(\tilde{\mathbf{H}}, \mathbf{0})$ as an *augmented instance* of $(\mathbf{H}, \mathbf{s})$. Note that for any solution $\tilde{\mathbf{e}} \in \mathbb{F}_{q^m}^n$ to the augmented instance, i.e.,

$$\tilde{\mathbf{H}}\tilde{\mathbf{e}}^\top = \mathbf{0} \quad \text{and} \quad w_R(\tilde{\mathbf{e}}) \leq r,$$

there exists a nonzero $u \in \mathbb{F}_{q^m}$ such that $\tilde{\mathbf{e}} - u \cdot \mathbf{y} \in \mathcal{C}$. Thus, $u^{-1}\tilde{\mathbf{e}}$ is a solution to the initial RSD instance $(\mathbf{H}, \mathbf{s})$.

Remark 2.1. Note that a solution $\mathbf{e}$ to the original instance $(\mathbf{H}, \mathbf{s})$ is also a solution to the augmented one, that is, $\tilde{\mathbf{H}}\mathbf{e}^\top = 0$. This is because $\mathbf{H}(\mathbf{y}-\mathbf{e})^\top = \mathbf{s}^\top - \mathbf{s}^\top = \mathbf{0}$. Thus, $\mathbf{y} - \mathbf{e} \in \mathcal{C}$ which implies that $\mathbf{e}$ belongs to the augment code $\tilde{\mathcal{C}}$.

The primary advantage of this approach is that a solution to the original instance $(\mathbf{H}, \mathbf{s})$ yields $q^m - 1$ solutions to the augmented instance: if $\tilde{\mathbf{e}}$ is a solution of $(\tilde{\mathbf{H}}, \mathbf{0})$, then so is every nonzero scalar multiple $\alpha\tilde{\mathbf{e}}$ for $\alpha \in \mathbb{F}_{q^m}^*$. This comes at the usually minor cost of increasing the code dimension by one, which leads to fewer parity-check equations and in turn less information available about the solution.

Gilbert-Varshamov Bound Generally, the difficulty of decoding increases with the rank weight r of the solution. As r increases, the searchspace grows increasing the hardness of the problem, while beyond a certain threshold, this trend reverses due to the higher likelihood of multiple solutions. This threshold is known as the Rank Gilbert–Varshamov (RGV) bound [28].

Define $S_r = \{\mathbf{e} \in \mathbb{F}_{q^m}^n : w_R(\mathbf{e}) = r\}$ for $r \geq 0$ to be the search space for rank weight r solutions. It holds $|S_0| = 1$ and for $r \geq 1$

$$|S_r| = \prod_{j=0}^{r-1} \frac{(q^m - q^j)(q^n - q^j)}{q^r - q^j},$$

by counting the number of rank r matrices in $\mathbb{F}_q^{m\times n}$, since each such vector in $\mathbb{F}_{q^m}^n$ of rank r can be seen as a matrix over $\mathbb{F}_q$ of the same rank under a fixed basis. Let $B_r = \bigcup_{i=0}^{r} S_i$, with $|B_r| = \sum_{i=0}^{r} |S_i|$. The Rank Gilbert-Varshamov (RGV) bound or simply GV-bound for a *linear* (n,k)*-code* $\mathcal{C}$ over $\mathbb{F}_{q^m}$ is the smallest integer r such that

$$q^{m(n-k)} \leq |B_r|.$$

Using the approximation $|B_r| \approx q^{(m+n-r)r}$, we obtain the RGV bound as the smallest r such that $m(n-k) \leq (m-r)r + nr$. We call this value r_{GV} [17]. Furthermore, random codes are known to asymptotically meet the RGV bound, i.e., a randomly chosen syndrome admits a solution of rank at most r when $r \geq r_{\mathrm{GV}}$.

3 Solving the Rank Syndrome Decoding Problem

In this section we give a structured overview of algorithms solving the RSD problem. The algorithmic strategies followed by these algorithms can be categorized into *combinatorial*, *algebraic* and *hybrid* approaches.

The main goal of this section is to provide unified cost estimates for all known algorithms solving RSD and lay the foundation for the following theoretical analysis and concrete comparison of those algorithms in Section 4.

We measure time complexity in terms of elementary multiplications over $\mathbb{F}_q$, and memory usage in terms of the number of stored $\mathbb{F}_q$ elements. In order to obtain bit-level estimates we account for every $\mathbb{F}_q$ multiplication with $\log^2 q$ bit-operations and for every $\mathbb{F}_q$ element with $\log q$ bits. Let $n_1 \leq n_2$ denote the dimensions of a matrix over $\mathbb{F}_q$. We account for Gaussian Elimination over $\mathbb{F}_q$ with a runtime complexity of $\mathcal{O}\left(n_1 n_2^{w-1}\right)$ elementary operation over $\mathbb{F}_q$, where $2 \leq w \leq 3$ is the matrix-multiplication constant. In the square case $n_1 \sim n_2$ this reduces to $\mathcal{O}(n_2^w)$. All over this section, we use the following notation:

- $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_m)$ to denote a $\mathbb{F}_q$-basis of $\mathbb{F}_{q^m}$,
- $\mathbf{e}$ to denote an error vector in $\mathbb{F}_{q^m}^n$ with $w_R(\mathbf{e}) = r$,
- $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_r)$ to refer to a basis of $\mathsf{Supp}(\mathbf{e})$, and
- $\mathbf{A} \in \mathbb{F}_q^{r\times n}$ to denote the matrix such that $\mathbf{e} = \boldsymbol{\theta} \cdot \mathbf{A}$.

3.1 Combinatorial Algorithms

Combinatorial algorithms for rank decoding typically follow a common framework, composed of two main steps: an *enumeration* phase, where subsets or candidates are enumerated, and a *linear system solving* phase, where the validity of each candidate is verified by solving a system of linear equations over $\mathbb{F}_q$. The total complexity of such algorithms can be generically expressed as:

$$\text{Total Cost} = \text{Number of Enumerated Candidates} \times \text{Cost per Linear System}.$$

Typically, the enumeration step has an exponential time complexity, while the linear solving step comes at a polynomial cost. In the subsequent analyses we follow this decomposition for every individual algorithm.

Basis Enumeration (BE) Let $(\mathbf{H}, \mathbf{s})$ be an instance of $\mathrm{RSD}(q, m, n, k, r)$. Remember that any $\mathbf{e} \in \mathbb{F}_{q^m}^n$ with a basis $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_r)$ of $\mathsf{Supp}(\mathbf{e})$ can be written as

$$\mathbf{e} = \boldsymbol{\theta} \cdot \mathbf{A},$$

for some $\mathbf{A} \in \mathbb{F}_q^{m \times n}$. Hence, $\mathbf{e}$ is a vector of n bilinear equations in the coordinates of $\boldsymbol{\theta}$ and $\mathbf{A}$. Thus, the substitution of $\boldsymbol{\theta}$ by any vector in $\mathbb{F}_{q^m}^r$ turns $\mathbf{e}$ into a vector of n linear equations in the coordinates of $\mathbf{A}$. Furthermore, the equation $\mathbf{H}\mathbf{e}^\top = \mathbf{s}^\top$ yields $(n-k)$ linear equations with coefficients in $\mathbb{F}_{q^m}$ in the nr coordinates of $\mathbf{A}$ which are in $\mathbb{F}_q$. These $(n-k)$ linear equations can be converted into $m(n-k)$ linear equations with coefficients in $\mathbb{F}_q$ by coefficient decomposition, i.e., by decomposing each equation via the $\mathbb{F}_q$ basis $\boldsymbol{\beta}$ and treating the coefficient of each β_i as one equation.

The naive BasisEnumeration consists on enumerating one basis $\boldsymbol{\theta}$ of each r-dimensional vector spaces in $\mathbb{F}_{q^m}$. Hence the number of bases to enumerate is the number of r-dimensional vector spaces in $\mathbb{F}_{q^m}$, which is given by the Gaussian coefficient

$$\begin{bmatrix} m \\ r \end{bmatrix}_q = \prod_{i=0}^{r-1} \frac{q^m - q^i}{q^r - q^i} \approx q^{(m-r)r}.$$

For each enumerated basis the resulting linear system of $m(n-k)$ equations in nr variables over $\mathbb{F}_q$ is solved, which is expected to yield a solution to RSD when we enumerate a basis of $\mathsf{Supp}(\mathbf{e})$ as long as $m(n-k) \geq nr$. Thus, the complexity of the naive enumeration is $\mathcal{O}(q^{(m-r)r} \cdot (nr)^w)$.

Chabaud and Stern [19] improved the naive enumeration approach by noting that the equivalent RSD instance given by $(\mathbf{H}, \tilde{\mathbf{s}})$, where $\tilde{\mathbf{s}} = \theta_1^{-1} \cdot \mathbf{s}$ admits a solution $\tilde{\mathbf{e}}$ with $1 \in \mathsf{Supp}(\tilde{\mathbf{e}})$. Indeed, setting $\tilde{\mathbf{e}} = \theta_1^{-1}\mathbf{e}$ where $\mathbf{e}$ is the solution to the original RSD instance $(\mathbf{H}, \mathbf{s})$, it follows that

$$\mathbf{H}\tilde{\mathbf{e}}^\top = \tilde{\mathbf{s}}^\top.$$

In addition, since $\mathsf{Supp}(\mathbf{e}) = \langle \theta_1, \ldots, \theta_r \rangle$, we obtain that $1 \in \mathsf{Supp}(\tilde{\mathbf{e}})$.

Similarly to the naive approach, for any basis $\tilde{\boldsymbol{\theta}} = (1, \tilde{\theta}_2, \ldots, \tilde{\theta}_r)$ of $\mathsf{Supp}(\tilde{\mathbf{e}})$ we can write

$$\tilde{\mathbf{e}} = \tilde{\boldsymbol{\theta}} \cdot \tilde{\mathbf{A}},$$

for some $\tilde{\mathbf{A}} \in \mathbb{F}_q^{r \times n}$. Chabaud and Stern propose to enumerate one basis for each r-dimensional vector space in $\mathbb{F}_{q^m}$ that contains 1. On the one hand, for each basis $\tilde{\boldsymbol{\theta}}$ of $\mathsf{Supp}(\tilde{\mathbf{e}})$, the product $\mathbf{H}\tilde{\mathbf{e}}^\top$ yields $m(n-k)$ equations in the nr coordinates of $\tilde{\mathbf{A}}$. On the other hand, since θ_1 is unknown, we express the coordinates of $\tilde{\mathbf{s}} = \theta_1^{-1} \cdot \mathbf{s}$ by writing θ_1^{-1} as a linear combination of a fixed $\mathbb{F}_q$-basis $\{\beta_1, \ldots, \beta_m\}$ of $\mathbb{F}_{q^m}$:

$$\theta_1^{-1} = z_1\beta_1 + \cdots + z_m\beta_m, \quad \text{with } z_1, \ldots, z_m \in \mathbb{F}_q.$$

This introduces m unknowns over $\mathbb{F}_q$. Substituting into the equation $\mathbf{H}\tilde{\mathbf{e}}^\top = \tilde{\mathbf{s}}^\top$ yields a linear system over $\mathbb{F}_q$ with $m(n-k)$ equations and $nr + m$ unknowns.

Complexity. The number of r-dimensional vector spaces in $\mathbb{F}_{q^m}$ containing 1 is given by

$$\begin{bmatrix} m-1 \\ r-1 \end{bmatrix}_q = \prod_{i=0}^{r-2} \frac{q^{m-1}-q^i}{q^{r-1}-q^i} \approx q^{(m-r)(r-1)}.$$

Thus, the complexity of this approach is

$$T_{\text{BasisEnumeration}} = \mathcal{O}\left((nr+m)^w \cdot q^{(m-r)(r-1)}\right), \tag{1}$$

field operations while the memory complexity is

$$M_{\text{BasisEnumeration}} = \mathcal{O}\left((n-k)(nmr+m^2)\right). \tag{2}$$

Ourivski–Johansson (OJ) [35] Let $(\mathbf{H}, \mathbf{s})$ be an instance of $\text{RSD}(q, m, n, k, r)$, and let $\tilde{\mathbf{e}}$ be a solution of the augmented $\text{RSD}(q, m, n, k+1, r)$ instance defined by $(\tilde{\mathbf{H}}, \mathbf{0})$ with $1 \in \mathsf{Supp}(\tilde{\mathbf{e}})$. Denote by $\tilde{\mathbf{G}} = [\mathbf{I}_{k+1} \mid \mathbf{R}]$, where $\mathbf{R} \in \mathbb{F}_{q^m}^{(k+1)\times(n-k-1)}$, the generator matrix of the code defined by $\tilde{\mathbf{H}}$ (see Section 2).

Let $\boldsymbol{\theta} = (1, \theta_2, \ldots, \theta_r) \in \mathbb{F}_{q^m}^r$ be an $\mathbb{F}_q$-basis of $\mathsf{Supp}(\tilde{\mathbf{e}})$, and $\mathbf{A} \in \mathbb{F}_q^{r\times n}$ be the full-rank matrix such that $\tilde{\mathbf{e}} = (1, \theta_2, \ldots, \theta_r) \cdot \mathbf{A}$. Then, it holds

$$(1, \theta_2, \ldots, \theta_r) \cdot \mathbf{A} \begin{bmatrix} \mathbf{R} \\ -\mathbf{I}_{n-k-1} \end{bmatrix} = \mathbf{0}. \tag{3}$$

In particular, for $1 \le N \le n-k-1$, it holds

$$(1, \theta_2, \ldots, \theta_r) \cdot \mathbf{A}_{k+1+N} \begin{bmatrix} \mathbf{R}_N \\ -\mathbf{I}_N \end{bmatrix} = \mathbf{0},$$

where $\mathbf{A}_{k+1+N}$ denotes the matrix formed by the first $k+1+N$ columns of $\mathbf{A}$, $\mathbf{R}_N$ the matrix formed by the first N columns of $\mathbf{R}$, and $\mathbf{I}_N$ the identity matrix of size N.

*First Strategy (*OJ1*).* The first algorithm proposed by Ourivski and Johansson builds over the following proposition

Proposition 3.1. *Let $\mathbf{C} \in \mathbb{F}_q^{r\times(k+1+N)}$ be a matrix such that*

1. *the last $r-1$ rows of $\mathbf{C}$ are a basis of the $(r-1)$-dimensional vector space spanned by the last $r-1$ rows of $\mathbf{A}_{k+1+N}$, and*
2. *the first row of $\mathbf{C}$ equals the first row of $\mathbf{A}_{k+1+N}$.*

Then, there is a basis $(1, \gamma_2, \ldots, \gamma_r) \in \mathbb{F}_{q^m}^r$ of $\mathsf{Supp}(\tilde{\mathbf{e}})$ such that $\tilde{\mathbf{e}} = (1, \gamma_2, \ldots, \gamma_r) \cdot \mathbf{C}$. In particular, it holds

$$(1, \gamma_2, \ldots, \gamma_r) \cdot \mathbf{C} \begin{bmatrix} \mathbf{R}_N \\ -\mathbf{I}_N \end{bmatrix} = \mathbf{0}.$$

Proof. By definition, there is $\mathbf{B} = \left(\begin{array}{c|c} 1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{B}' \end{array}\right) \in \mathbb{F}_q^{r\times r}$, where $\mathbf{B}' \in \mathbb{F}_q^{(r-1)\times(r-1)}$ is invertible, such that $\mathbf{C} = \mathbf{B}\mathbf{A}_{k+1+N}$. Finally, set $(1, \gamma_2, \ldots, \gamma_r) = (1, \theta_2, \ldots, \theta_r) \cdot \mathbf{B}^{-1}$. □

By Proposition 3.1, replacing the last $r-1$ rows of $\mathbf{C}$ by a guess of a basis of the vector space spanned by the last $r-1$ rows of $\mathbf{A}_{k+1+N}$ (which is of dimension $r-1$ for most matrices $\mathbf{A}$) yields a linear system of N equations with coefficients over $\mathbb{F}_{q^m}$ and unknowns given by $\gamma_2, \ldots, \gamma_r \in \mathbb{F}_{q^m}$ and the $a_1, \ldots, a_{k+1+N} \in \mathbb{F}_q$ entries in the first row of $\mathbf{A}_{k+1+N}$.

The above linear system is solved by unfolding it over $\mathbb{F}_q$, resulting in a linear system over $\mathbb{F}_q$ with mN equations in $(r-1)m+k+1+N$ unknowns. To ensure a unique solution, it suffices to take

$$N = \left\lceil \frac{(r-1)m+k+1}{m-1} \right\rceil . \tag{4}$$

*Second Strategy (*OJ2*).* As in BasisEnumeration, here we enumerate all possible bases of the form $(1, \gamma_2, \ldots, \gamma_r) \in \mathbb{F}_{q^m}^r$ of the support of the solution $\tilde{\mathbf{e}}$ of the augmented instance. Each basis, yields a linear system with N equations with coefficients over $\mathbb{F}_{q^m}$ and unknowns given by the entries of $\mathbf{A}_{k+1+N} \in \mathbb{F}_q^{r\times(k+1+N)}$. As before, to solve this system it is first unfolded over $\mathbb{F}_q$, which yields a linear system over $\mathbb{F}_q$ with mN equations and $r(k+1+N)$ unknowns. To avoid spurious solutions, we should take

$$N = \left\lceil \frac{(k+1)r}{m-r} \right\rceil . \tag{5}$$

Complexity. The cost of OJ1 is estimated as the cost of enumerating all $(r-1)$-dimensional subspaces of $\mathbb{F}_q^{k+1+N}$ and solving the corresponding linear system. The number of subspace candidates is

$$\begin{bmatrix} k+1+N \\ r-1 \end{bmatrix}_q \approx q^{(r-1)(k+N+2-r)}.$$

Thus, the complexity of this approach is

$$T_{\text{OJ1}} = \mathcal{O}\left(((r-1)m+k+N+1)^w q^{(r-1)(k+N+2-r)}\right).$$

while the memory complexity is

$$M_{\text{OJ1}} = \mathcal{O}(mN \cdot ((r-1)m+k+N+1)).$$

Note that as long as $k \leq 2(m-1)-r$ Equation (4) yields $N \leq r+1$. To the best of our knowledge, this condition on k is satisfied for any proposed RSD parameters in the literature (compare to Section 4.3). In turn the complexity of OJ1 rewrites as

$$\mathcal{O}\left((mr)^w \cdot q^{(r-1)(k+3)}\right).$$

The OJ2 strategy enumerates r-dimensional $\mathbb{F}_q$-subspaces of $\mathbb{F}_{q^m}$ that contain 1, so the number of candidates is

$$\begin{bmatrix} m-1 \\ r-1 \end{bmatrix}_q \approx q^{(m-r)(r-1)}.$$

For each candidate, it solves a linear system with mN equations and $(k+1+N)r$ variables over $\mathbb{F}_q$, costing $\mathcal{O}((k+1+N)^w \cdot r^w)$. Hence, the total runtime is

$$T_{\mathrm{OJ2}} = \mathcal{O}\left((k+1+N)^w \cdot r^w \cdot q^{(m-r)(r-1)}\right),$$

while the memory complexity is

$$M_{\mathrm{OJ2}} = \mathcal{O}(mN(k+1+N)r).$$

GRS. Let $(\mathbf{H}, \mathbf{s})$ be an instance of $\mathrm{RSD}(q, m, n, k, r)$. The algorithm proposed by Gaborit, Ruatta and Schrek (GRS) [27] is based on the following fact: If $\bar{\boldsymbol{\theta}} = (\bar{\theta}_1, \ldots, \bar{\theta}_{\bar{r}})$ a basis of a vector space containing $\mathsf{Supp}(\mathbf{e})$, where $\mathbf{e}$ is a solution to the instance $(\mathbf{H}, \mathbf{s})$, then

$$\mathbf{e} = \bar{\boldsymbol{\theta}} \cdot \bar{\mathbf{A}},$$

for some $\bar{\mathbf{A}} \in \mathbb{F}_q^{\bar{r} \times n}$. Additionally, the equation $\mathbf{H}\mathbf{e}^\top = \mathbf{s}^\top$ yields $m(n-k)$ linear equations over $\mathbb{F}_q$ in the entries of $\bar{\mathbf{A}}$. Thus, knowledge of such a basis $\bar{\boldsymbol{\theta}}$ can be used to compute $\mathbf{e}$ by solving a linear system with $m(n-k)$ equations and $\bar{r}n$ variables as long as $\bar{r}n \leq m(n-k)$.

Naive Approach. A naive application of the GRS algorithm enumerates one basis of each $\bar{r}$-dimensional vector space in $\mathbb{F}_{q^m}$. For a given basis $\bar{\boldsymbol{\theta}}$, it finds a solution to the corresponding linear system, if any, and builds the matrix $\bar{\mathbf{A}}$. Then, it checks whether $w_R(\mathbf{e}) = r$ with $\mathbf{e} = \bar{\boldsymbol{\theta}} \cdot \bar{\mathbf{A}}$ has rank r, and if not repeats with the next basis. We expect $w_R(\mathbf{e}) = r$ and, hence, the solution to be found, when the algorithm selects a basis $\bar{\boldsymbol{\theta}}$ of a vector space containing $\mathsf{Supp}(\mathbf{e})$. The probability that a randomly selected $\bar{r}$-dimensional vector space in $\mathbb{F}_{q^m}$ contains $\mathsf{Supp}(\mathbf{e})$ is

$$\frac{|\{E \in \mathsf{Gr}(\bar{r}, \mathbb{F}_{q^m}) \mid \mathsf{Supp}(\mathbf{e}) \subseteq E\}|}{|\mathsf{Gr}(\bar{r}, \mathbb{F}_{q^m})|} = \frac{\begin{bmatrix} m-r \\ \bar{r}-r \end{bmatrix}_q}{\begin{bmatrix} m \\ \bar{r} \end{bmatrix}_q} = \prod_{i=0}^{r-1} \frac{q^{\bar{r}} - q^i}{q^m - q^i} \approx q^{-r(m-\bar{r})}.$$

Complexity. The above probability grows in $\bar{r}$. Hence, we set $\bar{r}$ to be the maximum value such that $\bar{r}n \leq m(n-k)$, i.e., $\bar{r} = \lfloor \frac{m(n-k)}{n} \rfloor = m - \lceil \frac{km}{n} \rceil$, to maintain an overdetermined linear system. We expect to enumerate $q^{r\lceil \frac{km}{n} \rceil}$ spaces until one contains $\mathsf{Supp}(\mathbf{e})$. For each guess, the algorithm tries to solve a linear system with $m(n-k)$ equations and $(m - \lceil \frac{km}{n} \rceil)n$ variables. This yields a time complexity of $\mathcal{O}(q^{r\lceil \frac{km}{n} \rceil}(m - \lceil \frac{km}{n} \rceil)^w n^w)$, while the memory complexity is $\mathcal{O}(m(n-k)(m - \lceil \frac{km}{n} \rceil)n)$.

Approach on the augmented instance. Gaborit, Ruatta and Schrek suggested in [27] to improve the algorithm by enumerating vectors spaces containing the support of the solution $\tilde{\mathbf{e}}$ to the augmented RSD instance $(\tilde{\mathbf{H}}, \mathbf{0})$ (see Section 2). If $(\theta_1, \ldots, \theta_r)$ is a basis of $\mathsf{Supp}(\mathbf{e})$ and $\mathbf{e}$ a solution to the instance $(\mathbf{H}, \mathbf{s})$, then $\tilde{\mathbf{e}} = \theta_1^{-1}\mathbf{e}$ is a solution to the instance $(\tilde{\mathbf{H}}, \mathbf{0})$ with $1 \in \mathsf{Supp}(\tilde{\mathbf{e}})$.

For any basis $\tilde{\boldsymbol{\theta}} = (1, \tilde{\theta}_2, \ldots, \tilde{\theta}_{\bar{r}})$ of an $\bar{r}$-dimensional vector space that contains $\mathsf{Supp}(\tilde{\mathbf{e}})$, there is a matrix $\tilde{\mathbf{A}} \in \mathbb{F}_q^{\bar{r} \times n}$ such that $\tilde{\mathbf{e}} = \tilde{\boldsymbol{\theta}} \cdot \tilde{\mathbf{A}}$. Hence the equation $\tilde{\mathbf{H}}\tilde{\mathbf{e}}^\top = \mathbf{0}$ yields $m(n-k-1)$ linear equations in the $\bar{r}n$ entries of $\tilde{\mathbf{A}}$.

In this version of the GRS algorithm, it enumerates one basis of each $\bar{r}$-dimensional vector space in $\mathbb{F}_{q^m}$ that contains 1. For each enumerated basis $\tilde{\boldsymbol{\theta}}$, as in the naive approach, from the equation $\tilde{\mathbf{H}}\tilde{\mathbf{e}}^\top = \mathbf{0}$, we derive $m(n-k-1)$ linear equations equations in the $\bar{r}n$ unknown entries of $\tilde{\mathbf{A}}$. The probability that a randomly selected $\bar{r}$-dimensional vector space in $\mathbb{F}_{q^m}$ that contains 1 also contains $\mathsf{Supp}(\tilde{\mathbf{e}})$ is

$$\frac{|\{E \in \mathsf{Gr}(\bar{r}, \mathbb{F}_{q^m}) \mid \mathsf{Supp}(\tilde{\mathbf{e}}) \subseteq E\}|}{|\{E \in \mathsf{Gr}(\bar{r}, \mathbb{F}_{q^m}) \mid 1 \in E\}|} = \frac{\begin{bmatrix} m-r \\ \bar{r}-r \end{bmatrix}_q}{\begin{bmatrix} m-1 \\ \bar{r}-1 \end{bmatrix}_q} = \prod_{i=1}^{r-1} \frac{q^{\bar{r}} - q^i}{q^m - q^i} \approx q^{-(r-1)(m-\bar{r})}.$$

Complexity. To maintain an overdetermined linear system, $\bar{r}$ is set such that $\bar{r}n \leq m(n-k-1)$, i.e., $\bar{r} = \lfloor \frac{m(n-k-1)}{n} \rfloor = m - \lceil \frac{m(k+1)}{n} \rceil$. The enumeration part is expected to run in $q^{(r-1)(\lceil \frac{m(k+1)}{n} \rceil)}$ iterations. For each guess, the algorithm tries to solve a linear system with $m(n-k-1)$ equations and $(m - \lceil \frac{m(k+1)}{n} \rceil)n$ variables. This yields a time complexity of

$$T_{\mathrm{GRS}} = \mathcal{O}\left(m^w (n-k-1)^w q^{(r-1)\left\lceil \frac{m(k+1)}{n} \right\rceil} \right),$$

while the memory complexity is

$$M_{\mathrm{GRS}} = \mathcal{O}\left(m(n-k-1)\left(m - \left\lceil \frac{m(k+1)}{n} \right\rceil \right) n \right).$$

Improved GRS ($\mathbf{GRS_{im}}$) Aragon, Gaborit, Hauteville and Tillich [4] have shown that the complexity of the GRS algorithm is improved by not restricting the enumeration to spaces containing 1 in their support. Instead they enumerate all $\bar{r}$-dimensional linear subspaces in $\mathbb{F}_{q^m}$ while still working with the augmented instance. They show that the algorithm succeeds, whenever they find a vector space E that contains a subspace of the form $\mathsf{Supp}(\alpha\mathbf{e})$ for some $\alpha \in \mathbb{F}_{q^m}^*$. From such a space E we can retrieve the solution $\mathbf{e}$ as in GRS.

Let $(\mathbf{H}, \mathbf{s})$ be an instance of the RSD(q, m, n, k, r) problem with solution $\mathbf{e} \in \mathbb{F}_{q^m}^n$. As seen in Section 2, for each $\alpha \in \mathbb{F}_{q^m}^*$, the vector $\alpha\mathbf{e}$ is a solution of the augmented RSD instance $(\tilde{\mathbf{H}}, \mathbf{0})$. Now, for any basis $\boldsymbol{\theta}_\alpha$ of an $\bar{r}$-dimensional vector space containing $\mathsf{Supp}(\alpha\mathbf{e})$ there is a matrix $\mathbf{A}_\alpha \in \mathbb{F}_{q^m}^{\bar{r} \times n}$ such that $\alpha\mathbf{e} = \boldsymbol{\theta}_\alpha \cdot \mathbf{A}_\alpha$. Hence the equation $\tilde{\mathbf{H}}(\alpha\mathbf{e})^\top = \mathbf{0}$ yields $m(n-k-1)$ linear equations in the $\bar{r}n$

unknown entries of $\mathbf{A}_\alpha$. We expect to recover $\mathbf{A}_\alpha$ and α by solving that system as long as $\bar{r}n \leq m(n-k-1)$.

We denote by $\mathcal{E}$ the set of the subspaces of the form $\mathsf{Supp}(\alpha\mathbf{e})$, with $\alpha \in \mathbb{F}_{q^m}^*$. It holds that $|\mathcal{E}| = \frac{q^m-1}{q-1}$ with overwhelming probability [4]. Furthermore, in [4], the authors conjecture that, under the random choice of $E \in \mathsf{Gr}(\bar{r}, \mathbb{F}_{q^m})$, it holds

$$\Pr[\bigcup_{S\in\mathcal{E}} \{S \subseteq E\}] \approx |\mathcal{E}| \cdot \Pr[S_0 \subseteq E],$$

for a fixed $S_0 \in \mathcal{E}$. In addition, the authors experimentally verified that the above approximation holds. In that case it holds

$$\begin{aligned}\Pr[\bigcup_{S\in\mathcal{E}} \{S \subseteq E\}] &\approx \frac{q^m-1}{q-1}\Pr[S_0 \subseteq E] = \frac{q^m-1}{q-1}\frac{\begin{bmatrix} m-r \\ \bar{r}-r \end{bmatrix}_q}{\begin{bmatrix} m \\ \bar{r} \end{bmatrix}_q} = \frac{q^m-1}{q-1}\prod_{i=0}^{r-1}\frac{q^{\bar{r}}-q^i}{q^m-q^i} \\ &= \left(\sum_{j=0}^{m-1} q^j\right)\prod_{i=0}^{r-1}\frac{q^{\bar{r}}-q^i}{q^m-q^i} \leq \left(\sum_{j=0}^{m-1} q^j\right) q^{-r(m-\bar{r})} \leq q^{m-r(m-\bar{r})}\end{aligned}$$

Complexity. To maintain an overdetermined linear system, choose $\bar{r}$ so that $\bar{r}n \leq m(n-k-1)$, i.e. $\bar{r} = \lfloor \frac{m(n-k-1)}{n} \rfloor = m - \lceil \frac{m(k+1)}{n} \rceil$. Thus $m - \bar{r} = \lceil \frac{m(k+1)}{n} \rceil$, and the enumeration part is expected to run in $\mathcal{O}\left(q^{r(\lceil \frac{m(k+1)}{n} \rceil)-m}\right)$ iterations. For each guess, the algorithm tries to solve a linear system with $m(n-k-1)$ equations and $(m - \lceil \frac{m(k+1)}{n} \rceil)n$ variables. This yields a time complexity of

$$T_{\mathrm{GRS}_{\mathrm{im}}} = \mathcal{O}\left(m^w(n-k-1)^w q^{r(\lceil \frac{m(k+1)}{n} \rceil)-m}\right),$$

while the memory complexity is $M_{\mathrm{GRS}_{\mathrm{im}}} = M_{\mathrm{GRS}}$.

Guessing-enhanced GRS algorithm ($\mathbf{GRS_{ge}}$) D'Alconzo, Esser, Gangemi and Sanna [20] proposed a further enhancement to the improved GRS algorithm, incorporating a guessing subroutine. Essentially their improvement exploits that usually the optimal choice of $\bar{r}$ leads to *more* equations than unknowns in the resulting linear system, while it suffices to have the same amount of equations as unkowns.

Note that each known $\mathbb{F}_q$-entry of $\mathbf{B}$, where $\mathbf{e} = \boldsymbol{\beta} \cdot \mathbf{B}$ leads to an additional linear equation. The guessing-enhanced GRS algorithm now aims at guessing t entries of $\mathbf{B}$, which leads to a final system with $n\bar{r}$ unknowns and $(n-k-1)m+t$ equations. Therefore a choice of $\bar{r} = m - \lceil \frac{(k+1)m-t}{n} \rceil$ suffices to ensure the system is overdetermined. Since we expect q^t iterations until the guess is correct, we obtain a running time of

$$T_{\mathrm{GRS}_{\mathrm{ge}}} = \mathcal{O}\left(\min_{t\geq 0}\left(((n-k)m+t)^w \cdot \max\left(1,\, q^{r\left\lceil \frac{(k+1)m-t}{n}\right\rceil - m + t}\right)\right)\right),$$

while the memory requirement is

$$M_{\mathrm{GRS}_{\mathrm{ge}}} = \mathcal{O}\Big(\left\lfloor \tfrac{(n-k-1)m+t}{n}\right\rfloor\; n\left((n-k-1)m+t\right)\Big).$$

3.2 Algebraic Algorithms

The particular nature of the rank metric makes it a natural field for algebraic attacks. When directly applicable these attacks are largely independent of the field size q and the extension degree m. However, often these attacks require specific prerequisites, which, if not met, need to be obtained by a hybrid technique, that combines algebraic techniques with an enumeration subroutine. The complexity of this subroutine itself then depends on the field size q again. We first detail the pure algebraic approaches and their respective requirements, after which we detail the hybrid technique used in cases where those requirements are not met.

Early algebraic approaches. Levy-dit and Perret [36] proposed one of the first algebraic algorithms to solve the RSD problem. It builds on ideas by Ourivski–Johansson (OJ) and Chabaud–Stern (BasisEnumeration) .

Let $(\mathbf{H}, \mathbf{s})$ be an instance of $\mathrm{RSD}(q, m, n, k, r)$ and $(\tilde{\mathbf{H}}, \mathbf{0})$ a corresponding augmented instance. Let $\tilde{\mathbf{e}}$ be a solution to $(\tilde{\mathbf{H}}, \mathbf{0})$, and $\tilde{\boldsymbol{\theta}} = (1, \tilde{\theta}_2, \ldots, \tilde{\theta}_r)$ a basis of $\mathsf{Supp}(\tilde{\mathbf{e}})$. By Equation (3), we have that

$$\boldsymbol{\theta}(\mathbf{A}_1\mathbf{R} - \mathbf{A}_2) = \mathbf{0},$$

where $\mathbf{A} = [\mathbf{A}_1 \mid \mathbf{A}_2]$, $\mathbf{A}_1 \in \mathbb{F}_q^{r\times(k+1)}$, $\mathbf{A}_2 \in \mathbb{F}_q^{r\times(n-k-1)}$ are unknown, and $\mathbf{R} \in \mathbb{F}_{q^m}^{(k+1)\times(n-k-1)}$ is known. In addition,

$$\boldsymbol{\theta}[\mathbf{A}_1 \mid \mathbf{A}_2]\mathbf{H}^\top = u\mathbf{s}^\top,$$

for some unknown $u \in \mathbb{F}_{q^m}$, see Section 3.1 on BasisEnumeration. The two above equations yield a quadratic system over $\mathbb{F}_q$ with $2m(n-k)-1$ equations and $(n+m)r$ unknowns. The quadratic part of this system is bilinear in only $nr + m(r-1)$ unknowns over $\mathbb{F}_q$ (the entries of $[\mathbf{A}_1 \mid \mathbf{A}_2]$ and the coefficients of the unknown coordinates of $\boldsymbol{\theta}$).

To solve the system above, the authors proposed to use Gröbner basis solvers such as F4 [25]. The authors do not provide a full theoretical complexity analysis of their approach but it is expected to be outperformed by modern algebraic approaches, which we detail in the following.

Other early algebraic approaches to solve RSD include those designed to solve the MinRank problem, such as the Kipnis-Shamir and Minors algorithms, since an RSD instance can be efficiently reduced to an instance of the MinRank problem [24]. Hence, any MinRank attack can be used to solve RSD. However, these algorithms are also outperformed by modern algebraic algorithms for RSD.

Annulator Polynomial (AP). The AnnulatorPolynomial algorithm by Gaborit et al. [27] models RSD as a set of multivariate equations of degree q^{r+1} over the field extension $\mathbb{F}_{q^m}$. The model builds on the concept of q-polynomials [34], which we define in the following.

Definition 3.1 (q-polynomial). *Let P be a polynomial in $\mathbb{F}_{q^m}[X]$. We say that P is a q-polynomial of q-degree r if it is of the form*

$$P(X) = X^{q^r} + \sum_{i=0}^{r-1} p_i X^{q^i} \quad , \text{ where } p_i \in \mathbb{F}_{q^m}.$$

Proposition 3.2. *For any subspace E of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$ of dimension r there exists a unique monic q-polynomial P of q-degree r, such that for all $e \in E$, $P(e) = 0$. P is called the annulator polynomial of E.*

Proof. See [34, Theorem 8].

Let $(\mathbf{H}, \mathbf{s})$ be an RSD(q, m, n, k, r) instance with solution $\mathbf{e}$. Define $\mathbf{G}$ as a generator matrix of the code with parity-check matrix $\mathbf{H}$. Define $\mathbf{y} \in \mathbb{F}_{q^m}^n$ as a vector satisfying $\mathbf{H}\mathbf{y}^\top = \mathbf{s}^\top$. Then, it holds

$$\mathbf{e} = \mathbf{y} - \mathbf{u} \cdot \mathbf{G} \tag{6}$$

By Proposition 3.2, there exists a unique monic q-polynomial P of q-degree r, such that for all $e \in \mathsf{Supp}(\mathbf{e})$, $P(e) = 0$. Thus, for each $j = 1, \ldots, n$, it holds

$$\begin{aligned} 0 = P(e_j) = P(y_j - \langle \mathbf{u}, \mathbf{g}_j \rangle) &= (y_j - \langle \mathbf{u}, \mathbf{g}_j \rangle)^{q^r} + \sum_{i=0}^{r-1} p_i (y_j - \langle \mathbf{u}, \mathbf{g}_j \rangle)^{q^i} \\ &= y_j^{q^r} - \langle \mathbf{u}^{q^r}, \mathbf{g}_j^{q^r} \rangle + \sum_{i=0}^{r-1} p_i y_j^{q^i} - \langle p_i \mathbf{u}^{q^i}, \mathbf{g}_j^{q^i} \rangle), \end{aligned}$$

where $\mathbf{a}^{q^i}$ denotes the vector that results of raising each coordinate of $\mathbf{a} \in \mathbb{F}_{q^m}^k$ to the q^i-th power.

Linearization. By treating each coordinate of $\mathbf{u}^{q^r}$, the coefficients of p_i and each coordinate of the vectors $p_0\mathbf{u}, p_1\mathbf{u}^q, \ldots, p_{r-1}\mathbf{u}^{q^{r-1}}$ as unknowns, the above n equations are linear with a set $(r+1)(k+1)-1$ variables, namely:

- k variables from the coordinates of the vector $\mathbf{u}^{q^r}$.
- k variables per vector of the form $p_i\mathbf{u}^{q^i}$, for each $i = 0, \ldots, r-1$.
- The r variables $p_0, \ldots, p_{r-1}$.

Hence, in the case $n \geq (k+1)(r+1)-1$ we expect that a solution to the linear system yields a solution for $\mathbf{e}$ to the RSD instance $(\mathbf{H}, \mathbf{s})$. In such a case, the runtime complexity of this algorithm is $\mathcal{O}\left(n((r+1)(k+1)-1)^{w-1}\right)$ operations in $\mathbb{F}_{q^m}$.

MaxMinors (MM). Expanding on their earlier work [7], Bardet et al. [9] introduced the MaxMinors algorithm.

Let $(\mathbf{H}, \mathbf{s})$ be an RSD(q, m, n, k, r) instance with solution $\mathbf{e}$. If $(\tilde{\mathbf{H}}, \mathbf{0})$ is an augmented RSD$(q, m, n, k+1, r)$ instance of $(\mathbf{H}, \mathbf{s})$, then it holds that $\mathbf{e} \cdot \tilde{\mathbf{H}}^\top = \mathbf{0}$ (see Remark 2.1). Let $\boldsymbol{\theta} \in \mathbb{F}_{q^m}^r$ be a basis of $\mathsf{Supp}(\mathbf{e})$ and $\mathbf{A} \in \mathbb{F}_q^{r \times n}$ such that $\mathbf{e} = \boldsymbol{\theta} \cdot \mathbf{A}$. The MaxMinors modeling [9] builds from the following fact

$$\boldsymbol{\theta} \cdot \mathbf{A}\tilde{\mathbf{H}}^\top = \mathbf{0},$$

which implies $\boldsymbol{\theta}$ is an element in the kernel of the matrix $\mathbf{A}\tilde{\mathbf{H}}^\top \in \mathbb{F}_{q^m}^{r \times (n-k-1)}$. Thus, any r-sized minor [3] of $\mathbf{A}\tilde{\mathbf{H}}^\top$ is zero. By the Cauchy-Binet formula, each of these minors can be expressed as a sum of different products of one r-sized minor of $\mathbf{A}$ (which is unknown) and one r-sized minor of $\tilde{\mathbf{H}}$ (which can be efficiently computed). Hence, treating the r-sized minors of $\mathbf{A}$ as variables, we obtain for each r-sized minor of $\mathbf{A}\tilde{\mathbf{H}}^\top$ a homogeneous linear equation over $\mathbb{F}_{q^m}$ in those variables. Thus, in total one obtains a linear system with $\binom{n-k-1}{r}$ equations with coefficients in $\mathbb{F}_{q^m}$ and $\binom{n}{r}$ unknowns over $\mathbb{F}_q$ (since the unknowns are minors of a matrix with coefficients in $\mathbb{F}_q$). Each of these equations can be unfolded as m equations with coefficients in $\mathbb{F}_q$. Then, unfolding the whole system, we obtain a linear system with $m\binom{n-k-1}{r}$ homogeneous equations and $\binom{n}{r}$ unknowns over $\mathbb{F}_q$.

Note that the r-sized minors of any matrix $\mathbf{C} \in \mathbb{F}_q^{r \times m}$ with the same row-space of $\mathbf{A}$ also form a solution to the above linear system. In order to eliminate these multiple solutions one can specialize one (unknown) minor to 1. Thus, the system constructed by the MaxMinors algorithms admits a unique solution as long as

$$m\binom{n-k-1}{r} \geq \binom{n}{r} - 1. \tag{7}$$

When the above inequality is not satisfied one has to resort to the hybrid strategy which we describe in Section 3.2.

Complexity. To improve the running time of the MaxMinors algorithm one can puncture the code defining the RSD instance in p positions before running the algorithm, implying that its length is reduced from n to $n-p$. For RSD(q, m, n, k, r) instances satisfying inequality 7, the complexity of MaxMinors is

$$T_{\text{MaxMinors}} = \mathcal{O}\left(m\binom{n-p-k-1}{r}\binom{n-p}{r}^{w-1}\right), \tag{8}$$

where

$$p = \max\left\{i \in [0, \ldots, n] : m\binom{n-i-k-1}{r} \geq \binom{n-i}{r} - 1\right\},$$

while the amount of $\mathbb{F}_q$-elements to be stored is

$$M_{\text{MaxMinors}} = \mathcal{O}\left(m\binom{n-p-k-1}{r}\binom{n-p}{r}\right). \tag{9}$$

[3] An r-sized minor of a matrix $\mathbf{M}$ is the determinant of an $r \times r$ submatrix of $\mathbf{M}$.

SupportMinors (SM). The SupportMinors algorithm for RSD was introduced by Bardet et al. in [8] as an extension to the SupportMinors algorithm for the MinRank problem by Bardet et al. [9], building on similar concepts as the MaxMinors.

Given an RSD instance $(\mathbf{H}, \mathbf{s})$ with a unique solution $\mathbf{e}$. We know that $\mathbf{e} = \mathbf{y} + \mathbf{u} \cdot \mathbf{G}$, where $\mathbf{y} \in \mathbb{F}_{q^m}^n$ is any vector satisfying $\mathbf{H}\mathbf{y}^\top = \mathbf{s}^\top$, and $\mathbf{G}$ is a generator matrix of the code defined by the parity-check matrix $\mathbf{H}$. Furthermore, we have that for a basis $\boldsymbol{\theta}$ of $\mathsf{Supp}(\mathbf{e})$ there is a unique matrix $\mathbf{A}$ over $\mathbb{F}_q$ such that $\mathbf{e} = \boldsymbol{\theta} \cdot \mathbf{A}$. Thus,

$$\mathbf{y} + \mathbf{u} \cdot \mathbf{G} = \boldsymbol{\theta} \cdot \mathbf{A}.$$

The above equation implies that the vector $\mathbf{y} + \mathbf{u} \cdot \mathbf{G}$ lies in the $\mathbb{F}_{q^m}$-row space of the matrix $\mathbf{A}$. In orther words, each minor of size $r+1$ of the concatenated matrix

$$\begin{bmatrix} \mathbf{y} + \mathbf{u} \cdot \mathbf{G} \\ \mathbf{A} \end{bmatrix}$$

is zero. Each of those minors is a bilinear polynomial in the k coordinates $u_1, \ldots, u_k \in \mathbb{F}_{q^m}$ of $\mathbf{u}$ and the $\binom{n}{r}$ r-sized minors $c_T \in \mathbb{F}_q$ of $\mathbf{A}$. Hence we obtain a bilinear system of $\binom{n}{r+1}$ equations and $\binom{n}{r} + k$ variables, which we call the SupportMinors equations. The u_i variables are called the *linear variables* while the c_T variables are called the *minor variables*.

The above system is solved by first computing the MaxMinors equations, which are linear polynomials in the minor variables c_T. Then, use those equations to express $m\binom{n-k-1}{r}$ minor variables as linear combinations of the remaining $\binom{n}{r} - m\binom{n-k-1}{r}$. Next, substitute those expressions in the SupportMinors equations. Finally, multiply the resulting equation by all the monomials of degree at most $b-1$ in the u_i variables and solve the resulting system by linearization. The number of linearly independent equations of the linearized system is expected to be

$$\begin{aligned} N_b = &\sum_{i=1}^{k} \binom{n-i}{r}\binom{k+b-1-i}{b-1} - \binom{n-k-1}{r}\binom{k+b-1}{b} \\ &- (m-1)\sum_{i=1}^{b}(-1)^{i+1}\binom{k+b-i-1}{b-i}\binom{n-k-1}{r+i} \end{aligned}$$

while the number of variables is

$$M_b = \binom{k+b-1}{b}\left(\binom{n}{r} - m\binom{n-k-1}{r}\right).$$

see [8, Proposition 7].

Complexity. The SupportMinors algorithm succeeds in solving an instance of the RSD problem as long as $N_b \geq M_b - 1$. The estimated cost of this algorithm, in the number $\mathbb{F}_{q^m}$-operations, is then given by

$$T_{\text{SupportMinors}} = \mathcal{O}\left(N_b M_b^{w-1}\right), \tag{10}$$

while the number of $\mathbb{F}_{q^m}$ field elements to be stored is

$$M_{\text{SupportMinors}} = \mathcal{O}\left(N_b M_b\right). \tag{11}$$

As previously, if the prerequisites for SupportMinors are not satisfied, a fallback to the hybrid approach is required to reach the constraint $N_b \geq M_b - 1$.

Hybrid Approach The hybrid approach exploits that for any integer a, an RSD instance of parameters (q, m, n, k, r) reduces to solving (on expectation) q^{ar} RSD instances with parameters $(q, m, n-a, k-a, r)$. This strategy was already used in some algebraic algorithms, such as [9, 27], before being generalized and formalized by Bardet et al. in [8, Proposition 13]. It states the following general formula for the complexity of solving RSD

$$\text{Time-RSD}(q, m, n, k, r) = \min_{a \geq 0} \left(q^{ar} \cdot \text{Time-RSD}(q, m, n-a, k-a, r)\right),$$

where $\text{Time-RSD}(q, m, n, k, r)$ denotes the complexity of the RSD instance with parameters (q, m, n, k, r).

Let $(\mathbf{H}, \mathbf{s})$ be an $\text{RSD}(q, m, n, k, r)$ instance with unique solution $\mathbf{e}$. Let $\mathbf{G}$ be a generator matrix of the code defined by $\mathbf{H}$ and $\mathbf{y} \in \mathbb{F}_{q^m}^n$ a vector satisfying $\mathbf{H}\mathbf{y}^\top = \mathbf{s}^\top$, which can be efficiently computed. This implies the existence of a vector $\mathbf{u} \in \mathbb{F}_{q^m}^k$ such that

$$\mathbf{e} = \mathbf{y} - \mathbf{u} \cdot \mathbf{G}.$$

At its core, the hybrid approach iteratively samples invertible matrices $\mathbf{R} \in \mathbb{F}_q^{n \times n}$, until the last a coordinates of

$$\bar{\mathbf{e}} := \mathbf{e} \cdot \mathbf{R} = \underbrace{\mathbf{y} \cdot \mathbf{R}}_{:=\bar{\mathbf{y}}} - \mathbf{u} \cdot \underbrace{\mathbf{G}\mathbf{R}}_{:=\bar{\mathbf{G}}},$$

are equal to zero. Hence, the last a entries of $\bar{\mathbf{y}} - \mathbf{u} \cdot \bar{\mathbf{G}}$ yield a linear equations in the k entries of $\mathbf{u}$. Next, these linear equations are used to substitute a coordinates of $\mathbf{u}$ by an affine function of the remaining $k-a$ coordinates. After this substitution the equation above turns into the form

$$\mathbf{e}' = \mathbf{y}' - \mathbf{u}' \cdot \mathbf{G}', \tag{12}$$

where $\mathbf{u}'$ is the vector formed by $k-a$ coordinates of $\mathbf{u}$. The new low-weight vector $\mathbf{e}' \in \mathbb{F}_{q^m}^{n-a}$, which is formed by the first $n-a$ coordinates of $\bar{\mathbf{e}}$, remains unknown and $w_R(\mathbf{e}') = w_R(\mathbf{e})$, while $\mathbf{y}' \in \mathbb{F}_{q^m}^{n-a}$ and $\mathbf{G}' \in \mathbb{F}_{q^m}^{(k-a)\times(n-a)}$ are known. Finally, note that $(\mathbf{G}', \mathbf{y}')$ is an $\text{RD}(q, m, n-a, k-a, r)$ instance, which yields an equivalent $\text{RSD}(q, m, n-a, k-a, r)$ instance. Clearly, a solution to that instance yields a solution to the original RSD instance.

Remark 3.1. Put differently, the hybrid approach enumerates a vectors in the right kernel of the unknown matrix $\mathbf{A} \in \mathbb{F}_q^{r \times n}$ where $\mathbf{A}$ satisfies $\mathbf{e} = \boldsymbol{\theta} \cdot \mathbf{A}$ with $\boldsymbol{\theta} \in \mathbb{F}_{q^m}^r$ being a basis of the support of $\mathbf{e}$.

Following the logic of Remark 3.1, the probability to sample an invertible matrix $\mathbf{R} \in \mathbb{F}_q^{n \times n}$ such that the last a coordinates of $\mathbf{e} \cdot \mathbf{R}$ are zero is therefore q^{-ar}. This implies that, on expectation, we have to solve q^{ar} $\mathrm{RSD}(q, m, n-a, k-a, r)$ instances to find a solution to the original $\mathrm{RSD}(q, m, n, k, r)$ problem.

Hybrid Annulator Polynomial. In the particular case of the AnnulatorPolynomial algorithm described in Section 3.2, it is known that the optimal value of a in the hybrid approach is

$$a = \left\lceil \frac{(r+1)(k+1) - (n+1)}{r} \right\rceil \tag{13}$$

as long as $a \leq k$. In this case, the complexity of this algorithm is given by

$$\mathcal{O}\left(r^w k^w q^{r \left\lceil \frac{(r+1)(k+1)-(n+1)}{r} \right\rceil} \right). \tag{14}$$

Note that for MaxMinors and SupportMinors such an analytical analysis is not known and they therefore rely on numerical optimization of the hybrid parameter a.

4 Complexity Comparison

In this section we analyze and compare the complexities of different algorithms, using Section 3 as a basis. We start this comparison from an asymptotic perspective which simplifies the obtained runtime expressions from the previous section. This allows us to more clearly understand the impact of different problem parameters and isolate parameter regimes of particular interest. We then compare the simplified asymptotic exponents against the concrete complexity expressions from the previous section to evaluate how well those approximate the running times for cryptographic instances. Eventually, we discuss parameters suggested in the literature and provide concrete estimates.

4.1 Asymptotic Complexity

We now first derive simplified asymptotic expressions for the runtime statements that allow us to study in which regimes certain algorithms are favorable. Our analysis reveals that asymptotically the running time of many of the outlined algorithms converges, leading to only three substantially different asymptotic complexities to solve RSD.

Category	Algorithm	$\log_q T$	c in $T = q^{cn^2+o(n^2)}$
Enumeration	BasisEnumeration	$(m-r)(r-1)$	$(c_m - c_r)c_r$
	OJ2	$(m-r)(r-1)$	
Algebraic	OJ1	$(k+3)(r-1)$	$c_k c_r$
	AnnulatorPolynomial	$r\lceil\frac{(r+1)(k+1)-(n+1)}{r}\rceil$	
	MaxMinors	–	
	SupportMinors	–	
GRS	GRS	$(r-1)\lceil\frac{(k+1)m}{n}\rceil$	$c_k c_r c_m$
	ImprovedGRS	$(r\lceil\frac{(k+1)m}{n}\rceil - m)$	
	GuessingEnhancedGRS	$\max\left(0, (r\lceil\frac{(k+1)m-t}{n}\rceil - m + t)\right)$	

Table 1: Runtime exponents $\log_q T$ for T as in Section 3 and asymptotic exponents c in $T = q^{cn^2+o(n^2)}$ for different algorithms.

Simplified asymptotic exponents. In the following we assume that all problem parameters linearly depend on n, i.e., $m, k, r = \Theta(n)$. More precisely we assume that $m = c_m n$, $k = c_k n$ and $r = c_r n$ for constants c_m, c_k, c_r. We now derive for each algorithm its asymptotic exponent c in

$$T = q^{cn^2+o(n^2)}.$$

Table 1 summarizes all these exponents, which are derived from the runtime statements in Section 3 by disregarding polynomial factors. In this table we categorize algorithms by their asymptotic exponent into the three categories *Enumeration*, *Algebraic* and *GRS*. Note that this categorization is merely indicative and intended for easier reference, as the boundaries between categories are not strict.

BasisEnumeration *and* OJ2. Both algorithms obtain an asymptotic running time of $\tilde{\mathcal{O}}\left(q^{(m-r)(r-1)}\right)$, while the exponent rewrites as

$$(m-r)(r-1) = (c_m - c_r)(c_r - o(1))n^2 = (c_m - c_r)c_r n^2 + o(n^2).$$

As mentioned earlier OJ2 can be seen as a version of BasisEnumeration with improved polynomial factors, which explains that both algorithms obtain the same exponent.

AnnulatorPolynomial *and* OJ1. The complexity exponent of the first strategy of Ourivski-Johansson (OJ1) rewrites similarly to above as $c_k c_r$. For the hybrid AnnulatorPolynomial approach (see Equation (14)) we observe the same by noting that r in the exponent cancels, as the ceiling only affects second order terms. Note that the convergence of both running times to the same asymptotic exponent is no coincidence. Observe that for $r, k = \Theta(n)$ the optimal value of a in hybrid application of the AnnulatorPolynomial algorithm (see Equation (13)) converges to k. Following Remark 3.1, the hybrid technique

for $a = k$ enumerates k vectors in the right kernel of the matrix $\mathbf{A}$. The OJ1 algorithm, on the other hand, enumerates a matrix $\mathbf{B}$ with row-space as a fixed column submatrix of $\mathbf{A}$ with $k + N$ columns. Computing the right kernel of $\mathbf{B}$ yields $k + N - r$ vectors, which, once extended by zeros outside the selected columns, yield right-kernel vectors of $\mathbf{A}$. Eventually note that for $m, k, r = \Theta(n)$ Equation (4) yields $N \approx r$. Hence, both algorithms effectively enumerate the same type of linear dependencies, leading to identical asymptotic exponents.

MaxMinors. Recall, that the MaxMinors algorithm constructs a system of $m\binom{n-k-1}{r}$ equations and $\binom{n}{r}$ unknowns. However, in the asymptotics we have that this system is always underdetermined as $m\binom{n-k-1}{r} = o\left(\binom{n}{r}\right)$. Therefore the MaxMinors algorithm has to be instantiated via the hybrid technique with parameter a to guess enough kernel vectors (compare to Remark 3.1) such that $m\binom{n-k-1}{r} \geq \binom{n-a}{r}$. However, for $m, k, r = \Theta(n)$ this only holds for $a \sim k$ which implies a running time of $\tilde{\mathcal{O}}\left(q^{kr}\right)$ (compare to Hybrid in Section 3.2) which lies in $q^{c_k c_r n^2 + o(n^2)}$.

SupportMinors. Deriving a closed form for the runtime exponent of the SupportMinors algorithm is not straightforward due to the expressions for the number of linearly independent equations N_b and number of variables M_b and the corresponding optimization of b. However, analytical analysis shows that similar to the MaxMinors case, for the asymptotic modeling of $k, m, r = \Theta(n)$ it holds $N_b < M_b$ for any choice of b. Moreover when applying the hybrid technique we require similarly $a \sim k$ leading to the same complexity as in the MaxMinors case. This fact is evidenced in the concrete estimates shown in later Figure 3 and Figure 4.

GRS family. The standard GRS complexity follows directly from observing that the ceiling in the exponent only affects second order terms. Furthermore the $-m$ term in the complexity exponent of ImprovedGRS also only affects the $o(n^2)$ term of the exponent. An analogous argument holds for the optimization parameter t in the GuessingEnhancedGRS algorithm: For t to affect the asymptotic running time we have to model $t = c_t n^2$ for some constant c_t, in which case the runtime exponent c becomes

$$c = c_k c_r c_m - c_t c_r + c_t,$$

which is minimized for $c_t = 0$, since $c_r < 1$. In other words, as intuition suggests, all algorithms of the GRS family achieve the same asymptotic runtime exponent.

Asymptotic regimes. In summary, Table 1 shows that, up to second-order terms, all known algorithms collapse into three asymptotic regimes. This observation reveals fundamental dependencies between instance parameters and optimal algorithms. In the following, we discuss these regimes and the effect of each individual parameter in more detail. We analyze algorithmic performance for rank

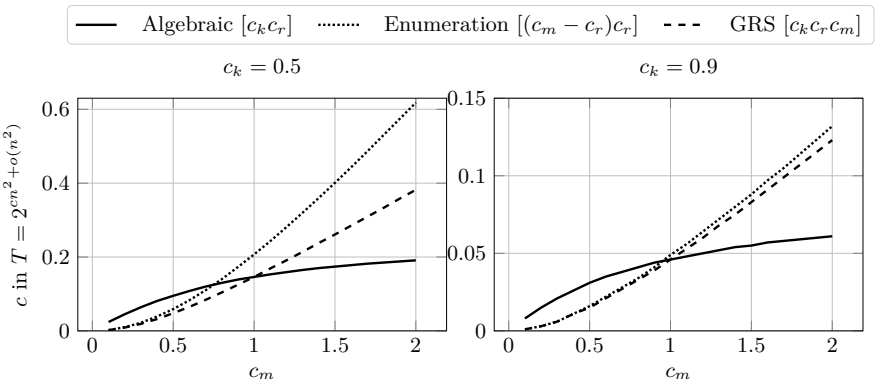

Fig. 1: Asymptotic exponent for $c_r = c_{r_{\mathrm{GV}}}$ and $c_k \in \{0.5, 0.9\}$.

weights staying below the GV-bound. Recall that the GV-bound $r_{\mathrm{GV}} = c_{r_{\mathrm{GV}}} n$ is the positive solution of $m(n-k) = (m-r)r + nr$ (compare to Section 2) and, hence $c_{r_{\mathrm{GV}}}$ satisfies

$$c_{r_{\mathrm{GV}}} = \frac{1 + c_m - \sqrt{4c_k c_m + (c_m - 1)^2}}{2}.$$

The extension degree m. Comparing the asymptotic exponent of algebraic approaches reaching $c_k c_r$ and the GRS family of $c_k c_r c_m$, we observe that algebraic attacks remain inferior to GRS-style algorithms whenever we have $m < n$, or analogously $c_m < 1$, while the opposite is the case for $m > n$ ($c_m > 1$). At the transition point $m = n$ ($c_m = 1$) all algorithms of these two categories obtain the same runtime exponent. Comparing those running times to the enumeration category with a runtime exponent of $(c_m - c_r)c_r$, we observe that this exponent regardless of the parameter choices remains inferior to the algorithms of the GRS family with both running times converging for rates close to one. This is due to the fact that for $c_k \to 1$ we have $c_{r_{\mathrm{GV}}} \to 0$ and, hence $c_r \to 0$, since $c_r \leq c_{r_{\mathrm{GV}}}$.

In Figure 1 we illustrate the runtime exponents as a function of c_m for $c_r = c_{r_{\mathrm{GV}}}$ fixed to the GV-bound.

The code dimension k. We now consider the running time of the algorithms as a function of the code dimension k or, more precisely, of the rate c_k. For now we fix $c_r = c_{r_{\mathrm{GV}}}$ to the GV-bound, but results remain similar if adjusted to smaller values. In Figure 2 we illustrate the runtime exponents as a function of the code rate c_k for $c_m = \frac{1}{2}$ (left) and $c_m = 2$ (right). We observe that the runtime exponents of the GRS family as well as of the algebraic approaches maximize around rate $c_k = \frac{1}{2}$, with a slight tilt to the left: For $c_m = \frac{1}{2}$, the GRS and algebraic algorithms both maximize at $c_k = 0.455$, while enumeration approaches maximize at $c_k = 0.375$. Furthermore regardless of the choice of c_m and c_r the complexity exponents of GRS and algebraic algorithms maximize at

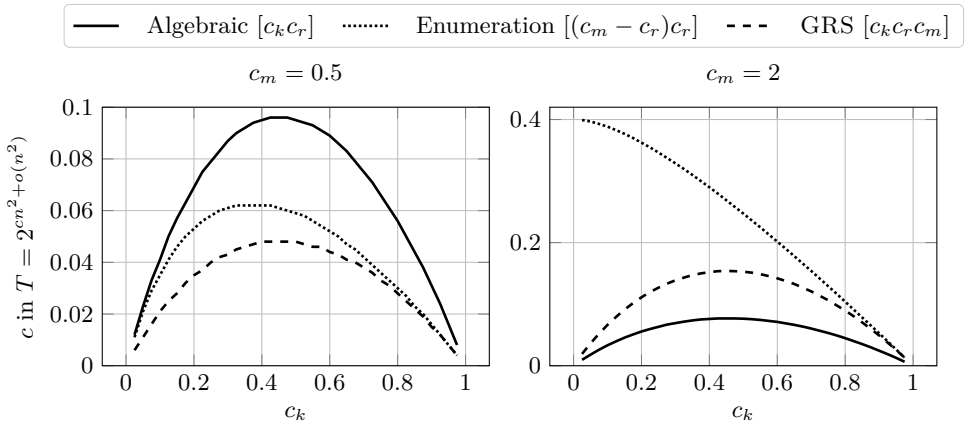

Fig. 2: Asymptotic exponent for $c_r = c_{r_{\mathrm{GV}}}$ and $c_m \in \{0.5, 2\}$.

the same c_k inside the interval $[0.445, 0.5]$. In case of the enumeration approaches, for growing c_m they tend to maximize at rates closer to zero, while for $c_m \geq 2$ the exponent maximizes at $c_k = 0$ (corresponding to $k = o(n)$).

The rank weight r. For the GRS class of algorithms and algebraic approaches, the runtime exponents behave linearly in the rank weight c_r, while the basis enumeration has a slight quadratic dependence. However, as expected, the runtime exponents of all algorithms are maximized at $c_r = c_{r_{\mathrm{GV}}}$.

The field size q. From an asymptotic perspective the field size q only introduces an equal scaling of all asymptotic exponents. Hence, while it increases the overall complexity, it does not affect the relation between the exponents.

Overall, the above analysis indicates that the runtime exponents of all algorithms grow with the extension degree m. However, for the specific case of $m = n$, algebraic approaches and GRS-inspired algorithms converge. Furthermore, for any fixed c_m, the exponents of these algorithms — similar to those of algorithms for the syndrome decoding problem in the Hamming metric — reach their maximum at the same rate around $c_k = \frac{1}{2}$. We therefore propose to follow the metric of *worst-case decoding exponents*, analogous to the Hamming metric case. The worst-case decoding exponent $\tilde{c}$ of an algorithm is defined as the runtime exponent maximized over c_k for $c_r = c_{r_{\mathrm{GV}}}$. Fixing $c_m = 1$ and $q = 2$ we obtain for GRS inspired algorithms and the algebraic techniques

$$\tilde{c}_{\mathrm{GRS}} = \tilde{c}_{\mathrm{Algebraic}} = 0.1481 \quad \text{at} \quad c_k = 0.445,$$

while for the enumeration techniques we get

$$\tilde{c}_{\mathrm{Enumeration}} = 0.25 \quad \text{at} \quad c_k = 0.25.$$

4.2 Asymptotic vs. Concrete Complexity

We now compare the simplified exponents from the previous section against the concrete runtime expressions from Section 3. The goal is to understand how precisely those exponents match the behavior of the algorithms for different sized parameters and to understand where respective differences originate from. Throughout this section we fix $q = 2$.

Comparison for different rates. In Figure 3 we illustrate the concrete runtime exponents for all algorithms [4] computed as $\frac{\log T}{n^2}$ for T being the concrete runtime expressions derived in Section 3. As comparison we also plot the asymptotic exponents c in $T = 2^{cn^2+o(n^2)}$ from the previous section (see Table 1). In this comparison we fix the extension degree to $c_m = 1$ and the rank weight to $c_r = c_{r_{\mathrm{GV}}}$ and then plot the exponents as functions of c_k. On the left we use $n = 80$, which leads to instances with roughly 250 bit complexity around $c_k = 1/2$. We clearly observe that the running times of those algorithm match their asymptotic behavior, while polynomial factors still lead to smaller deviations. For comparison we provide on the right in same figure a plot for $n = 200$ illustrating full convergence of the practical exponents towards their asymptotic versions.

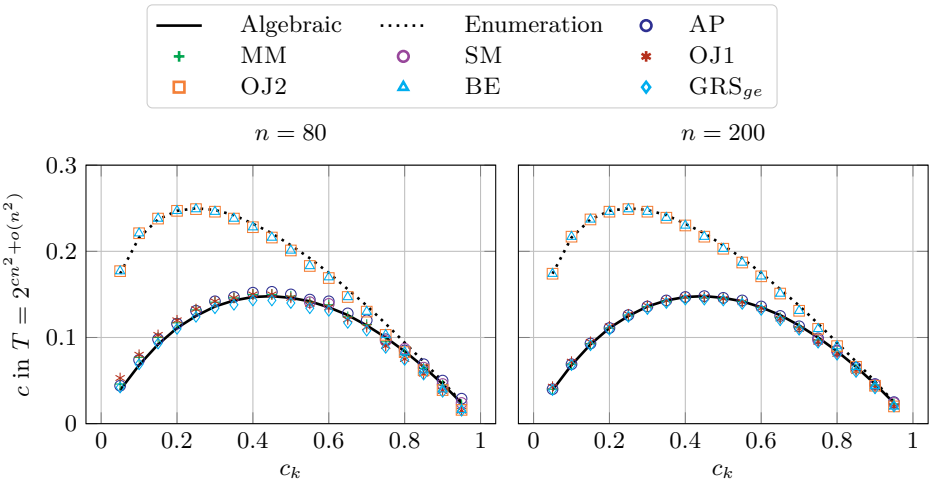

Fig. 3: Asymptomatic exponents vs concrete estimates for $c_m = 1$ and $c_r = c_{r_{\mathrm{GV}}}$.

Comparison for different weights. So far, we investigated the running times mostly for $c_r = c_{r_{\mathrm{GV}}}$, which is commonly considered the hardest case. However,

[4] For clarity we only include the latest improvement of the GRS inspired algorithms, as their complexity exponents are rather similar.

in practice, instances may deviate from that choice. In particular, instances with significantly smaller r (and thus smaller c_r) have been proposed. We therefore now examine the validity of the asymptotic exponents in these regimes.

In Figure 4 (left) we illustrate the asymptotic and concrete runtime exponents (for $n = 150$) of the different algorithms for $c_m = 1$ and $c_k = 1/2$ as a function of $c_r \in (0, c_{r_{\mathrm{GV}}}]$. We observe that the runtime exponents generally match the asymptotic prediction, except for the MaxMinors and SupportMinors algebraic approaches, which fall below their asymptotic values for smaller weights $c_r \leq \frac{c_{r_{\mathrm{GV}}}}{2}$.

As discussed earlier, MaxMinors as well as SupportMinors rely on the hybrid technique to compensate for the underdetermined nature of their linear system, introducing a guessing parameter a. For small r, the number of equations and unknowns in the original systems become closer, implying that fewer guesses must be made. Consequently, the optimal a deviates from its asymptotic prediction k, and the asymptotic estimate becomes less accurate. Even though, this effect vanishes in the asymptotic, for medium-sized concrete parameters it remains visible for small choices of r. On the right of Figure 4, we plot a/k as a function of the weight r for both algorithms, where a_{MM} (resp. a_{SM}) represents the optimal parameter a of the hybrid approach applied to MaxMinors (resp. SupportMinors). For very small r, the fallback to the hybrid approach becomes unnecessary, leading to the largest deviations. As r increases, a rapidly approaches k, and around $\frac{c_{r_{\mathrm{GV}}}}{2} \approx 0.15$, $a/k \approx 0.8$, resulting in a close match between concrete and asymptotic complexities.

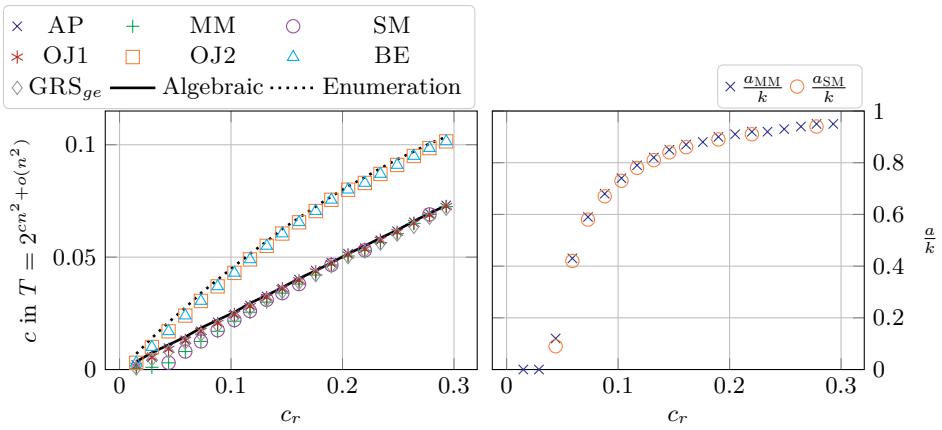

Fig. 4: Asymptotic exponents vs. concrete estimates as $c_r \to c_{r_{\mathrm{GV}}}$.

4.3 Concrete Complexity and Suggested Parameters

In this section, we investigate the impact of different algorithms on practical parameters. A compilation of parameter sets collected from the literature is summarized in Table 2. In particular, we consider dRANKula [1], RVDC [13], ROLLO [31], RQC [32], BG23 [16], RYDE [5,6], FEN24 [26], and DSD [17]. In what follows, we analyze the resistance of these parameter sets against particular algorithms and derive potential rationales for their selection. Table 2 lists, for each parameter set, the best combinatorial and algebraic attacks together with their corresponding bit-complexity estimates.

	Parameters				Combinatorial		Algebraic		
	m	n	k	r	algorithm	time	algorithm	time	a (hybrid)
dRANKula-256	127	118	48	7	GRS_{im}	270	MM	68	0
dRANKula-192	97	82	42	5	GRS_{im}	181	MM	47	0
dRANKula-128	67	57	27	5	GRS_{im}	119	MM	42	0
RVDC-96	29	22	11	7	OJ1	87	MM	79	8
RVDC-125	31	26	13	8	OJ1	113	MM	107	10
RVDC-196	41	34	17	10	OJ1	179	MM	173	14
RVDC-252	47	38	19	12	OJ1	238	MM	231	16
ROLLO-I-128	79	94	47	5	GRS_{im}	149	MM	50	0
ROLLO-I-192	89	106	53	6	GRS_{im}	211	MM	61	0
ROLLO-I-256	113	134	67	7	GRS_{im}	318	SM	127	5
ROLLO-II-128	83	298	149	5	GRS_{im}	154	MM	66	0
ROLLO-II-192	107	302	151	6	GRS_{im}	244	MM	78	0
ROLLO-II-256	127	314	157	7	GRS_{im}	349	SM	113	0
ROLLO-III-128	101	94	47	5	GRS_{im}	183	MM	49	0
ROLLO-III-192	107	118	59	6	GRS_{im}	248	MM	62	0
ROLLO-III-256	131	134	67	7	GRS_{im}	364	SM	107	2
RQC-I	97	134	67	5	GRS_{im}	178	MM	54	0
RQC-II	107	202	101	6	GRS_{im}	249	MM	71	0
RQC-III	137	262	131	7	GRS_{ge}	380	MM	108	3
BG23	31	30	15	9	GRS_{im}	139	MM	139	12
DSD-I	53	53	45	4	GRS_{im}	148	MM	169	37
DSD-II	79	75	67	4	OJ1	220	MM	254	58
DSD-III	97	95	87	4	OJ1	281	MM	331	77
RYDE-I	31	33	15	10	GRS_{ge}	138	MM	153	12
RYDE-II	37	41	18	13	GRS_{ge}	209	MM	237	15
RYDE-III	43	47	18	17	GRS_{im}	283	MM	308	15
RYDE-r2-I	53	53	45	4	GRS_{im}	148	MM	169	37
RYDE-r2-III	61	61	51	5	GRS_{im}	217	MM	245	44
RYDE-r2-V	61	57	44	7	OJ1	287	MM	301	39

Table 2: Concrete bit complexity estimates for different schemes using $w = 2$ in the estimation. All schemes use $q = 2$.

The impact of MaxMinors. Historically, the design of the MaxMinors algorithm has invalidated the security claims of many suggested parameters, previously considered secure. For example consider any instantiations of dRANKula, ROLLO or RQC, which suffered significantly reduced security ranging to full breaks of the suggested parameters. At the time, this led to the elimination of all rank based candidates from the NIST PQC standardization effort.

Following the asymptotic analysis from Section 4.1 one might expect the GRS style algorithms to perform better, as it holds $m < n$, i.e., $c_m < 1$, for all of these instances (compare to Table 1). However, recall that the asymptotic analysis assumes that $r = c_r n$ for constant c_r, requiring MaxMinors to fallback to the hybrid approach for guessing variables. On the other hand the design rationale for the mentioned schemes, which rely on efficient decoding algorithms for the chosen rank metric codes, moves them theoretically into the regime $r = \Theta(\sqrt{n})$. We have already seen in the previous section that smaller choices of r might affect the necessary fallback to the hybrid algorithm and, hence, the validity of asymptotic analysis. We show in following that technically the $r = \Theta(\sqrt{n})$ regime does not invalidate the asymptotic analysis, i.e., it requires the same full fallback to the hybrid approach with parameter $a \sim k$, leading to a complexity scaling as $q^{c_k c_r n^2} = q^{\Theta(n^{3/2})}$.

Still, for the practical parameters following this rationale we do not observe this fallback to the hybrid algorithm. Notice the hybrid parameter being zero for almost all affected parameters in Table 2. We find that this is an artifact of the particularly small constants chosen with respect to r. In fact for almost all affected parameters it holds $r < \log n$ which leads to a quasi-polynomial running time of MaxMinors as we show in the following.

MaxMinors *without Hybrid.* Recall, that as long as

$$m\binom{n-k-1}{r} \geq \binom{n}{r},$$

the linear system constructed by MaxMinors is overdetermined, and no hybrid application, i.e., no guessing is required. As already discussed in the asymptotic analysis in Section 4.1, for $r = \Theta(n)$ this inequality is never satisfied. While this might behave differently for smaller $r \ll n$, we show in the following that any $r = \omega(\log n)$ leads (asymptotically) to a full fallback to the hybrid algorithm with parameter $a \sim k$. In particular in this case, using the approximation $\binom{n}{r} \approx \frac{n^r}{r!}$ the above inequality rewrites as

$$m\left(\frac{n-k-1}{n}\right)^r \geq 1 \quad \Leftrightarrow \quad m \gtrsim (1-c_k)^{-r} \quad \Leftrightarrow \quad \frac{\log n}{r} \gtrsim -\log(1-c_k),$$

where $c_k = k/n$ assuming $m = c_m n$ with c_m (at most) constant and $a \gtrsim b$ iff $a \geq b \pm o(1)$. Since, for constant rates c_k, the right side of this inequality is constant, for any $r = \omega(\log n)$ the inequality can never be satisfied, as the left hand side converges to zero. On the other hand for $r = c'_r \log n$ for constant c'_r the inequality rewrites to

$$c_k \lesssim 1 - 2^{-1/c'_r}.$$

Since for the parameters of interest we have (with one slight deviation) $c'_r \leq 1$, this inequality becomes $c_k \leq 1/2$, which is satisfied for all parameters (but one, with $c_k = 0.51$). This explains why for those instances no hybrid application is necessary. Furthermore note that in those cases, i.e., $r = \Theta(\log n)$ the size of the system is quasi-polynomial as $\mathcal{O}(n^{\log n})$ and, hence, is the running time of MaxMinors, which leads to the significant gains we observe in Table 2.

Hybrid and combinatorial algorithms. Even though the hybrid technique, when formalized, was stated as a general technique applicable to any algorithm [8], the technique originates from its application to algebraic algorithms and has mostly been used in that context since.

From our concrete formulas we find that an application of the hybrid technique to GRS style algorithms can yield significant improvements over plain GRS in the regime $m > n$. This becomes especially apparent for instances using $r = r_{\mathrm{GV}}$ around the Gilbert-Varshamov bound. We illustrate the effect of the hybrid application to GRS style algorithms in Table 3.

	(25,9)	(28,9)	(31,10)	(34,10)	(37,11)	(40,11)	(43,11)	(46,12)	(49,12)	(52,12)
GRS	105	116	138	155	180	199	207	249	259	275
Hybrid GRS	105	116	138	155	180	**181**	**184**	**201**	**202**	**205**

Table 3: Concrete bit complexity estimates using $w = 2$ for GRS and hybrid application to GRS for different (m, r) values with RYDE's (round 1) choice of $n = 33$, $k = 15$.

For algorithms of the enumeration category we do not observe any improvement using the hybrid approach. This is because these algorithms do not benefit from the reduced dimension of the $\mathrm{RSD}(q, m, n-a, k-a, r)$ instance solved for each guess of the hybrid technique.

Secure RSD parameters. RSD instantiations that have been broken in the past were mostly relying on small choices for r. However, for large enough choices of r no efficient algorithms to solve RSD are known. Furthermore, as shown in section Section 4.1 for parameters using a rank weight equal to the Gilbert-Varshamov bound, i.e., $r = r_{\mathrm{GV}}$, all known algorithms have a complexity that is exponential in n^2.

Furthermore, even the regime of $r = \Theta(\sqrt{n})$ does not admit efficient algorithms. While here, algebraic approaches become subexponential in n^2, there complexity remains of order $q^{\mathcal{O}(n^{3/2})}$, as we have shown. However, for practical parameters caution is necessary with respect to this choice as the regime $r = \Theta(\log n)$ leads to quasi-polynomial algorithms.

Generally, Section 4.1 shows that the complexity of known algorithms maximizes around the common choice of rate $c_k = 1/2$. We note that RYDE round 2

parameters as well as those of the dual support decomposition (DSD) deviate from this choice. This is because those schemes focus on minimizing the witness size inside the used proof system which is about

$$(r-1)m + r(n-r).$$

Since this value does not depend on the code dimension k larger k, which leads to smaller rank weight r_{GV} is more optimal for those schemes. However, technically these schemes still follow a well justified rationale by using $m = n$ and $r = r_{\mathrm{GV}}$.

Overall, we find that the concrete estimates we provide do not invalidate the security of any further schemes. However, this was expected, since the particular estimates of this work, or more precisely their implementation in the CryptographicEstimators library module, have been the very selection rationale for parameters such as those of the RYDE signature scheme.

References

1. Al Shehhi, H., Bellini, E., Borba, F., Caullery, F., Manazno, M., Mateu, V.: An IND-CCA transformation of a McEliece like cryptosystem over rank metric. In: Submitted to ACNS 2019 (2018)
2. Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., Kelsey, J., Liu, Y.K., Miller, C., Moody, D., Peralta, R., et al.: Status report on the second round of the nist post-quantum cryptography standardization process. US Department of Commerce, NIST **2**, 69 (2020)
3. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3) (2015)
4. Aragon, N., Gaborit, P., Hauteville, A., Tillich, J.: A new algorithm for solving the rank syndrome decoding problem. In: IEEE International Symposium on Information Theory, ISIT. pp. 2421–2425 (2018)
5. Aragon, N., Magali Bardet, L.B., Chi-Domínguez, J.J., Dyseryn, V., Thibauld Feneuil, P.G., Joux, A., Rivain, M., Tillich, J.P., Vinçotte, A.: Ryde specification (2025), https://pqc-ryde.org/assets/downloads/ryde_specification_v2.1.0.pdf
6. Aragon, N., Magali Bardet, L.B., Chi-Domínguez, J.J., Dyseryn, V., Thibauld Feneuil, P.G., Joux, A., Rivain, M., Tillich, J.P., Vinçotte, A.: RYDE specification (2025), https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/ryde-spec-web.pdf
7. Bardet, M., Briaud, P., Bros, M., Gaborit, P., Neiger, V., Ruatta, O., Tillich, J.P.: An algebraic attack on rank metric code-based cryptosystems. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020. pp. 64–93. Springer International Publishing, Cham (2020)
8. Bardet, M., Briaud, P., Bros, M., Gaborit, P., Tillich, J.P.: Revisiting algebraic attacks on minrank and on the rank decoding problem. Des. Codes Cryptography **91**(11), 3671–3707 (Jul 2023). https://doi.org/10.1007/s10623-023-01265-x, https://doi.org/10.1007/s10623-023-01265-x
9. Bardet, M., Bros, M., Cabarcas, D., Gaborit, P., Perlner, R., Smith-Tone, D., Tillich, J.P., Verbel, J.: Improvements of algebraic attacks for solving the rank decoding and minrank problems. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020. pp. 507–536. Springer International Publishing, Cham (2020)

10. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. pp. 10–24 (2016). https://doi.org/10.1137/1.9781611974331.ch2
11. Becker, A., Gama, N., Joux, A.: Speeding-up lattice sieving without increasing the memory, using sub-quadratic nearest neighbor search. Cryptology ePrint Archive, Report 2015/522 (2015), https://eprint.iacr.org/2015/522
12. Becker, A., Joux, A., May, A., Meurer, A.: Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. pp. 520–536 (2012). https://doi.org/10.1007/978-3-642-29011-4_31
13. Bellini, E., Caullery, F., Gaborit, P., Manzano, M., Mateu, V.: Improved veron identification and signature schemes in the rank metric. In: 2019 IEEE International Symposium on Information Theory (ISIT). pp. 1872–1876 (2019). https://doi.org/10.1109/ISIT.2019.8849585
14. Bellini, E., Makarim, R.H., Sanna, C., Verbel, J.: An estimator for the hardness of the mq problem. In: Batina, L., Daemen, J. (eds.) Progress in Cryptology - AFRICACRYPT 2022. pp. 323–347. Springer Nature Switzerland, Cham (2022)
15. Bidoux, L., Feneuil, T., Gaborit, P., Neveu, R., Rivain, M.: Dual support decomposition in the head: Shorter signatures from rank sd and minrank. In: Chung, K.M., Sasaki, Y. (eds.) Advances in Cryptology – ASIACRYPT 2024. pp. 38–69. Springer Nature Singapore, Singapore (2025)
16. Bidoux, L., Gaborit, P.: Compact post-quantum signatures from proofs of knowledge leveraging structure for the pkp, sd and rsd problems. In: El Hajji, S., Mesnager, S., Souidi, E.M. (eds.) Codes, Cryptology and Information Security. pp. 10–42. Springer Nature Switzerland, Cham (2023)
17. Bidoux, L., Feneuil, T., Gaborit, P., Neveu, R., Rivain, M.: Dual support decomposition in the head: Shorter signatures from rank SD and MinRank. Cryptology ePrint Archive, Paper 2024/541 (2024), https://eprint.iacr.org/2024/541
18. Both, L., May, A.: Decoding linear codes with high error rate and its impact for LPN security. pp. 25–46 (2018). https://doi.org/10.1007/978-3-319-79063-3_2
19. Chabaud, F., Stern, J.: The cryptographic security of the syndrome decoding problem for rank distance codes. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 368–381. Springer (1996)
20. D'Alconzo, G., Esser, A., Gangemi, A., Sanna, C.: Sneaking up the ranks: Partial key exposure attacks on rank-based schemes. Cryptology ePrint Archive, Paper 2024/2070 (2024), https://eprint.iacr.org/2024/2070
21. Ducas, L., Esser, A., Etinski, S., Kirshanova, E.: Asymptotics and improvements of sieving for codes. pp. 151–180 (2024). https://doi.org/10.1007/978-3-031-58754-2_6
22. Esser, A., Bellini, E.: Syndrome decoding estimator. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) Public-Key Cryptography – PKC 2022. pp. 112–141. Springer International Publishing, Cham (2022)
23. Esser, A., Zweydinger, F.: New time-memory trade-offs for subset sum: Improving ISD in theory and practice. pp. 360–390 (2023). https://doi.org/10.1007/978-3-031-30589-4_13
24. Faugère, J.C., Levy-Dit-Vehel, F., Perret, L.: Cryptanalysis of minrank. In: Annual International Cryptology Conference. pp. 280–296. Springer (2008)
25. Faugére, J.C.: A new efficient algorithm for computing gröbner bases (f4). Journal of Pure and Applied Algebra **139**(1), 61–88 (1999). https://doi.org/https://doi.org/10.1016/S0022-4049(99)00005-5, https://www.sciencedirect.com/science/article/pii/S0022404999000055

26. Feneuil, T.: Building mpcith-based signatures from mq, minrank, and rank sd. In: Pöpper, C., Batina, L. (eds.) Applied Cryptography and Network Security. pp. 403–431. Springer Nature Switzerland, Cham (2024)
27. Gaborit, P., Ruatta, O., Schrek, J.: On the complexity of the rank syndrome decoding problem. IEEE Transactions on Information Theory **62**(2), 1006–1019 (2016)
28. Loidreau, P.: Properties of codes in rank metric. CoRR **abs/cs/0610057** (2006), http://arxiv.org/abs/cs/0610057
29. May, A., Meurer, A., Thomae, E.: Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. pp. 107–124 (2011). https://doi.org/10.1007/978-3-642-25385-0_6
30. May, A., Ozerov, I.: On computing nearest neighbors with applications to decoding of binary linear codes. pp. 203–228 (2015). https://doi.org/10.1007/978-3-662-46800-5_9
31. Melchor, C.A., Aragon, N., Bardet, M., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C.: ROLLO - Rank-Ouroboros, LAKE & LOCKER, https://pqc-rollo.org/doc/rollo-submission_2019-08-24.zip
32. Melchor, C.A., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Zémor, G.: Rank Quasi-Cyclic (RQC) - Second round version (2019), https://pqc-rqc.org/doc/rqc-submission_2019-04-10.zip
33. Micciancio, D., Voulgaris, P.: Faster exponential time algorithms for the shortest vector problem. pp. 1468–1480 (2010). https://doi.org/10.1137/1.9781611973075.119
34. Ore, O.: On a special class of polynomials. Transactions of the American Mathematical Society **35**(3), 559–584 (1933)
35. Ourivski, A.V., Johansson, T.: New technique for decoding codes in the rank metric and its cryptography applications. Problems of Information Transmission **38**(3), 237–246 (2002)
36. Levy-dit Vehel, F., Perret, L.: Algebraic decoding of rank metric codes. Talk at the Special Semester on Gröbner Bases - Workshop D1 pp. 1–19 (2006), https://www.ricam.oeaw.ac.at/specsem/srs/groeb/download/Levy.pdf
37. Wang, X., Liu, M., Tian, C., Bi, J.: Improved nguyen-vidick heuristic sieve algorithm for shortest vector problem. In: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security. pp. 1–9 (2011)